The Network
1. Milestone

Frank Steiler
DHBW Stuttgart / Staffordshire University
Student Number: 13005490d
Contact: frank@steiler.eu

March 21, 2014

Contents

Chapter 1

Introduction

With the rise of the Internet and its increased accessibility more and more people started using this engineering marvel. With its help the whole mankind can stay connected, even if there are hundreds of miles away from each other. Over years and decades the way to handle this virtual wonderworld did change rapidly: Was it first the interaction via electronic mail staying connected got a complete new dimension when the first generation of so called ”social networks” started to launch. *Myspace* and *Facebook* showed how popular and important it is with people to stay in contact. On top of that *Twitter* created a quick way to share news and information. Social media and *Twitter* especially played an important role within several important political events in the 21st century. For example the networks were used to organize events during the Arab Spring [How+11].

2012 *Facebook* went public and had the largest valuation for a new public company with $104 billion[Ger12]. Followed in 2013 by *Twitter*, now even listed on the *NYSE*. After its first day as a public company *Twitter* was valued a little over $31 billion [BBC13].

All these facts show the high acceptance of social media and their massive potential on the money market. Although it seems hard to built a new network from the scratch and to impose it. I believe there is many resentment against established networks that it is worth to do research and planning of a new web based social network. Following this document will try to initiate discussion and justify the the necessessity to introduce *The Network*.

Chapter 2

Market Analysis

The statistics about spreading of social networks are impressive. A survey by the Pew Research Center showed that 67% of US citizens are using social network. Users are nearly equally distributed over all ages, genders and races [DB13]. Facebook itself had 1.06 billion monthly user and 618 million daily user all over the world as of December 2012 [Inc13]. From figure 1 it can be retrieved that the amount was constantly increasing over the year 2012. It appears that the market does not have any potential for a really big growth in the near future, since nearly every possible user already has got a well connected social media account. So where should be any need in developing a new social network? The reasons are going to be discussed in the following sections.

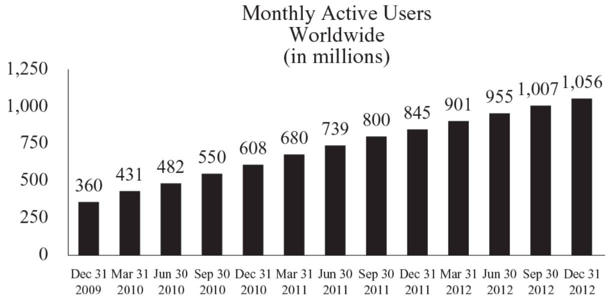
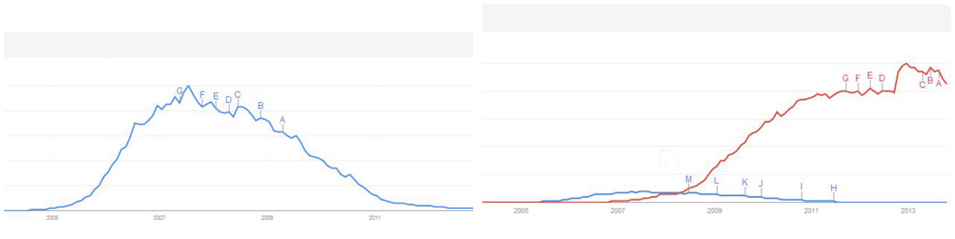


Figure 1: Monthly active users of *Facebook*

2.1 Lifecycle of social media

In 2007 *Myspace* was the most popular social network, and nobody thought that within a few years the former *king* could loose its crown. From figure 2a it can be retrieved that the interest in this network reached its climax by 2008. But then *Facebook* took over the lead and started to outclass *Myspace*. As Dube [Dub13] stated it is possible that *Facebook* –currently 'king of the hill'– could possible loose it's head start within the next couple of years to another competitor, just like it happened years before. This theory is supported by a recent study by researchers from Princeton. Their paper uses a statistical model which also fitted to the decline of *Myspace*. It states that *Facebook* will loose 80% of their users between 2015 and 2017 [CS14]. As shown in figure 3 the so called *irSIR* and *SIR* model matches very good



(a) *Myspace* (blue) 2004-2011

(b) *Myspace* (blue) & *Facebook* (red) 2005-2014

Figure 2: Interest on Social Networks over time according to *Google Trends*

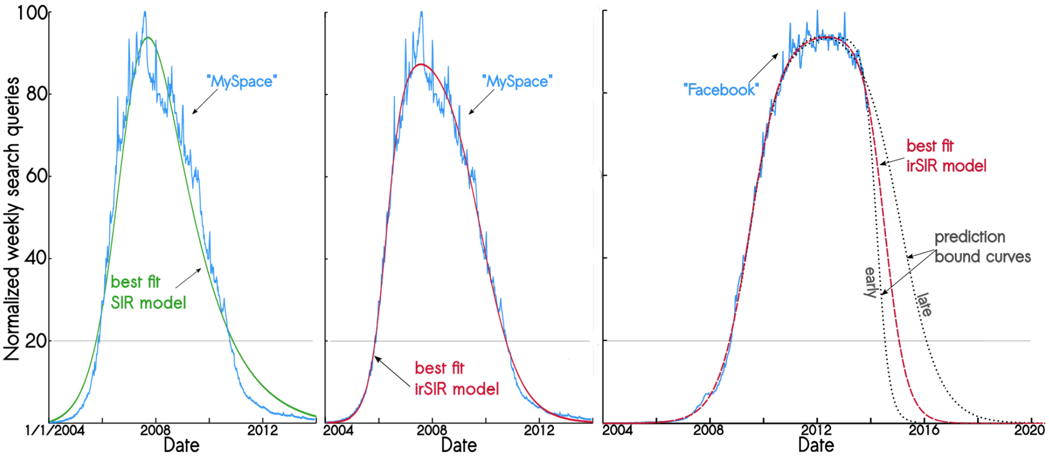


Figure 3: Princeton model applied to *Mypace* and *Facebook*

to the history of *Myspace*. If we adopt the curve to *Facebook*'s current development there is an early and late border for the predicted end of the network. Of course such a significant progress doesn't happen without any reason. Such reasons are currently heavily discussed by critics among the undisputed criticism to a lax handling of user data. The next chapters will focus on these points.

2.2 Aging of the user basis

Facebook's amount of users is constantly increasing since its release many years ago. But not only the amount of active users is important for the continued existence of a popular social network. It is also important to be attractive for younger user. That is currently a very big problem of the long-established social networks. A recent report of *iStrategyLabs* based on *Facebook*'s *Social Advertising* platform showed that the number of users between the age of 13 and 17 decreased by 25% since 2011. Also the age group of 18 up to 24 years old lost nearly 8% of their users [Sau14] . The biggest growth of users on *Facebook* has been in the group of people who are older than 55 years. On top of that there are 6.7 million fewer people in these demographics. These people got rid of their accounts and signed out of the social network. As Neal [Nea14] says, 'there is nothing cool about having parents and grandparents liking pictures of your friends.'. Furthermore he says that the younger generations are using newer messaging services like *Instagram*, *Snapchat* or *WhatsApp* more likely than a big social network, because of an easy usage and an increased privacy of messages in opposite to the public postings at *Facebook* [Nea13].

2.3 Overwhelming Advertisement

Of course it is very expensive to run a platform like *Facebook*. Since the business model of *Facebook* is based on a free usage for the end-user it was important to make money by selling personalized advertisement. Besides showing traditional banner alongside or on top of the page *Facebook* introduced *sponsored posts*. These are posts of paying customer, mainly companies, which are shown at the front page like every status of the friends of a user. But they appear even if the user doesn't *like* the advertising page.

Many user are enraged and complain about to only see ads [Slo13]. This is resulting in loosing users because of dissatisfaction. The *MIT Technology Review* states another problem connected with adverts. It shows that the overall revenue of advertisement per user is constantly decreasing. *Facebook* is only able to compensate this by increasing user count. Since this model is currently declining *Facebook* will struggle with lower revenues but constant maintenance costs [Wol12]. Concluding *Facebook* is not only loosing users but also revenues, a combination that can kill every business.

2.4 Conclusion

Summing up all of the facts leads to one conclusion: The market situation is going to change and *Facebook* will probably loose its lead. But that only will happen if there is going to be a better alternative. All the negative items mentioned above have to be improved to a new network giving a real alternative to current services. By considering all these aspects this project can get an enhanced and widely accepted new social platform that will grow for years and produce big revenue for its owner.

Chapter 3

Software Requirement Specification

Within the next sections the functional requirements of *The Network*, a web based social network, will be outlined. The project is going to be developed as part of the *Web Programming with Servlets and Java Server Pages* lecture held at the Staffordshire University 2014.

3.1 Product Purpose

The development of *The Network* should take account of the known problems of competitors. The outcome of this project should be a durable, easy-to-use and well designed social network. On top of that it should be competitive with existing services and based on the platform independent technologies *Java Server Pages*, *Java Servlets* and *Java*.

3.1.1 Definitions

To better understand the following sections it is important to define several terms. These definitions are shown in table 1 on the next page.

3.1.2 Obligatory Requirements

The fulfillment of the following criteria is mandatory:

- Every user can create an own profile.
- Every user can create an own fanpage.

Term	Description
Normal user	The representation of a person on the social network
Fanpage	The representation of an interest group, a club or a company. This page can be administrated by a single person or a whole department.

Table 1: Definition of specific terms

- Every fanpage can be connected to one user.
- There are admin accounts which can delete user and fanpages.
- Ever admin account can be connected to one user.
- Only the user can delete the connection to a fanpage or admin account.
- A user can follow a fanpage.
- A user can ask another user to be a friend. This *friendship* is only established after the other user accepts the friend request.
- Every user can delete the friendship with a user or un-follow a fanpage.
- Every user can publish a status post. He has the choice to post it public (for everyone) or private (only for friends).
- Every fanpage can only publish public status posts.
- The user can read all posts of friends and followed pages.
- The user can read all public posts of all users and pages.
- The user can comment on every post he is allowed to read.
- A fanpage can only comment on posts published by itself.
- The user can up-/down-vote every comment and status he is allowed to read.
- The user can only send messages to friends.
- The user gets notified if there are new messages, comments on posts he is following or a new friend request.

- Every user can change/delete its profile.
- Every fanpage can change/delete its page.
- The software needs to be programmed using GNU General Public License 3 (GPLv3).

3.1.3 Optional Requirements

An optional functionality is the development of an application programming interface (API), which provides a secure connection between applications and the server without the web interface. Within the project it is optional to prove the functionality of this interface by developing a java-based desktop application. This application should use XML-based communication and needs to have a professional standard graphical user interface (GUI).

3.1.4 Non-Requirements

The fulfillment of the following criteria is **not** part of the project:

- A fanpage cannot access any profile or fanpage.
- A fanpage cannot send or receive messages.
- A fanpage cannot see any status of users and therefor it won't be able to comment, down- or up-vote it.
- The admin account will not be able to post, comment, read, vote or manipulate any post or comment of any user.
- The admin account cannot read any messages.
- The admin account cannot send or receive any messages.
- An admin account cannot be created within the web or application interface.
- No one can upload photos besides the profile photo.
- Signing up or paying for a premium account is not considered within this version.

3.2 User

The key to develop a good application is the user. If the application is not focused on the user group it won't get accepted. The designated user group is going to be defined in the following section.

3.2.1 User characteristics

An open social network is only useful for an user if his friends, people he is interested in or organizations he wants to keep track of are part of the network. That's why the project does not have a specific user group, but needs to cover all possible users. If we focus on technical acceptance there is still a big diversity within the society based on the age. That's why our focus target group for a new launched social network is going to be the younger generations (Under 30). But we have to keep in mind that while the project is growing the user group will definitely expand to all age groups. So the design of the product needs to be as simple and intuitive as possible. To keep it simple the user interface (UI) needs to be clean. On top of that the range of functions is going to be reduced to the minimum so nobody is going to be put off because of an overwhelming amount of options.

3.2.2 User types

All analysis lead to 6 user types specified in table 2 on the following page.

3.3 Use-cases

Use-cases are an important step to define the project in a detailed way. The UML use case diagram which will give an overview for the project can be found in appendix A on page 136. Every use case is going to be described according to Cockburn [Coc01].

3.3.1 Manage friends & followed pages use-case

The Manage friends & followed pages use-case describes how a user can manage his connections within the network. The user is able to add and remove friends and fanpages. The UML use case diagram for this use case can be found in appendix A.1 on page 137.

Usertype	Description
Guest user	This is a user, which is not subscribed to the network. He will not have the possibility to explore the network unless he signs up for the service.
Normal free user	This user is a normal person exploring the social network. He does not pay for the use of the network, but there are going to be advertisements on every page.
Normal premium user	This user is a normal person exploring the social network. He pays for the use of the network, so there are not going to be any advertisements.
Free fanpage admin	This user administrates a fanpage, where he represents a society, company or interest group. He does not pay for the use of the network, but there are going to be advertisements on every page.
Premium fanpage admin	This user administrates a fanpage, where he represents a society, company or interest group. He pays for the use of the network, so there are not going to be any advertisements.
Admin user	This user administrates the network. He is able to view statistics and delete users or fanpages.

Table 2: Specification of user types within *The Network*

Use case: Show all friends

Table 3: Use case: Show all friends

Description:	A user wants an overview of all his friends.
Actors:	• User • Friend / Page Following Management Service
Pre-Conditions:	• User is logged in.
Post-Conditions:	**Success guarantees:** • The user sees a list of all his friends.
Normal flow:	1. User clicks on the ”Friends” entry in the header bar. 2. The systems collects all his friends. 3. All friends are listed on the next page.

Use case: Show all followed pages

Table 4: Use case: Show all followed pages

Description:	A user wants an overview of all pages he is following.
Actors:	• User • Friend / Page Following Management Service

Continued on next page

Table 4 – continued from previous page

Pre-Conditions:	• User is logged in.
Post-Conditions:	**Success guarantees:** • The user sees a list of all pages he is following.
Normal flow:	1. User clicks on the "Fanpages" entry in the header bar. 2. The systems collects all his followed pages. 3. All followed pages are listed on the next page.

Use case: Follow a page

Table 5: Use case: Follow a page

Description:	If a user is interested in a fanpage and wants to keep track of it he will follow this page.
Actors:	• User • Friend / Page Following Management Service
Pre-Conditions:	• User is not following the page. • User has found and opened the homepage of the fanpage.

Continued on next page

Table 5 – continued from previous page

Post-Conditions:	**Success guarantees:** • System adds the fanpage to the list of followed pages of the user.
Normal flow:	1. User clicks on the ”Follow page” button. 2. The systems adds the fanpage to the social profile of the user. 3. The system reloads the fanpage, indicating the following of the page by replacing the ”Follow page” button with an ”Unfollow page” button.

Use case: Add a friend

Table 6: Use case: Add a friend

Description:	If a user (A) is interested in another person (B) and wants to stay in contact with the other user (B) he will ask for a virtual friendship.
Actors:	• User (A) • Friend / Page Following Management Service

Continued on next page

Table 6 – continued from previous page

Pre-Conditions:	• User (A) is no friend of the other user (B). • User (A) has not added the other user (B) in the past as a friend. • User (A) has found and opened the profile of the other user (B).
Post-Conditions:	**Success guarantees:** • System stores the friend request. • The added user (B) receives a friend request.
Normal flow:	1. User (A) clicks on the ”Add as friend” button on the profile page of the other user (B). 2. The systems stores the request. 3. The added user (B) receives a friend request. 4. The system reloads the profile page, indicating the open friend request by replacing the ”Add as friend” button with a ”Request pending” button.

Use case: Accept a friend

Table 7: Use case: Accept a friend

Description:	If a user (B) is interested in a virtual friendship with another user (A), the other user (A) has to accept this request if he wants to be a friend.

Continued on next page

Table 7 – continued from previous page

Actors:	• User (A) • Friend / Page Following Management Service
Pre-Conditions:	• Other user (B) is no friend of the user. • Other user (B) has sent a request to the user (A).
Post-Conditions:	**Success guarantees:** • Friendship added to social profile of both user. • The other user (B) gets notified that he is now a friend of the user (A).
Normal flow:	1. User (A) clicks on the "Friends" menu item. 2. User (A) clicks on the "Accept" button next to the profile of the other user (B). 3. The systems stores the change of the friendship status. 4. The systems sets a notification flag, to notify the other user (B). 5. The system reloads the page, indicating the successful adding of the friend by removing the request and adding the user (B) to the "Friends" list.

Use case: Reject a friend

Table 8: Use case: Reject a friend

Description:	If a user (B) is interested in a virtual friendship with another user (A), the other user (A) has to reject this request if he does not want to be a friend with the user (B).
Actors:	• User (A) • Friend / Page Following Management Service
Pre-Conditions:	• Other user (B) is no friend of the user (A). • Other user (B) has sent a request to the user (A).
Post-Conditions:	**Success guarantees:** • Friend request gets rejected. • The other user (B) is not able to resend a friend request to user (A).

Continued on next page

Table 8 – continued from previous page

Normal flow:	1. User (A) clicks on the ”Friends” menu item. 2. User (A) clicks on the ”Reject” button next to the profile of the other user (B). 3. The systems stores the change of the friendship status. 4. The system reloads the page, indicating the successful rejection of the friend by removing the request.

Use case: Stop following a page

Table 9: Use case: Stop following a page

Description:	If a user is no longer interested in a fanpage, he will stop following this page.
Actors:	• User • Friend / Page Following Management Service
Pre-Conditions:	• User follows the fanpage. • User found the fanpage in the fanpage overview or found the fanpage itself.
Post-Conditions:	**Success guarantees:** • Fanpage gets removed from the social profile of the user.

Continued on next page

Table 9 – continued from previous page

Normal flow:	1. User clicks on the "Unfollow" button next to the page within the list or on the fanpage itself. 2. The system deletes the fanpage from the social profile of the user. 3. The system reloads the page indicating the success by removing the fanpage from the fan page overview list or replacing the "Unfollow" button with a "Follow page" button.

Use case: Remove a friend

Table 10: Use case: Remove a friend

Description:	If a user (A) does not want to be a friend with another user (B) anymore, he will delete this user (B)
Actors:	• User (A) • Friend / Page Following Management Service
Pre-Conditions:	• User (A) is friend of the other user (B). • User (A) found the user (B) in the friends overview or found the profile page of the user (B).

Continued on next page

Table 10 – continued from previous page

Post-Conditions:	**Success guarantees:** • Both users get removed from the social profile of each other.
Normal flow:	1. User clicks on the "Remove friend" button next to the profile within the list or on the profile page itself. 2. The system deletes both users from each others profile. 3. The system relaods the page indicating the success by removing the user (B) from the friends overview list of user (A) or replacing the "Remove friend" button with a "Add friend" button.

3.3.2 Access control use-case

The Login use-case describes how existing and new users can access the social network. The UML use case diagram for this use case can be found in appendix A.2 on page 138.

Use case: Log-in

Table 11: Use case: Sign-in

Description:	If a user wants to use the network he needs to sign into the network.

Continued on next page

Table 11 – continued from previous page

Actors:	• User • Page admin • Admin • User Management Service
Pre-Conditions:	• User is already signed up for the social network. • User is currently not signed in.
Post-Conditions:	**Success guarantees:** • System logs the user into his account.
Normal flow:	1. User enters his credential and clicks the ”Log in” button. 2. The systems validates the credentials. 3. The system forwards the user to his homepage depending on the type of user (Normal user homepage, Fanpage admin homepage or Admin homepage).
Alternative flow:	If the provided credentials are not valid the system is going to redirect the user back to the log in page where he will be able to re-enter the right credentials.

Use case: Sign-up as normal user

Table 12: Use case: Sign-up as normal user

Description:	If a guest user wants to join and use the social network he needs to sign up.
Actors:	• Guest • User Management Service
Pre-Conditions:	• The guest is not signed up for the social network.
Post-Conditions:	**Success guarantees:** • The system creates a new user and logs him into his account.
Normal flow:	1. User enters the requested details about himself and clicks on the "Sign up as a normal user" button 2. The systems creates a new account. 3. The systems invokes the log in function to log the user into his account 4. The system forwards the user to his homepage for a normal user.
Alternative flow:	If the user enters invalid information or not enough information the system will redirect him to the log in page to fill or correct the missing or wrong information.

Use case: Sign-up as fanpage admin

Table 13: Use case: Sign-up as fanpage admin

Description:	If a interest group or company wants to join and use the social network they need to sign up.
Actors:	• Guest • User Management Service
Pre-Conditions:	• The guest is not signed up for the social network.
Post-Conditions:	**Success guarantees:** • The system creates a new fanpage and logs the user into his account.
Normal flow:	1. User enters the requested details about the group or company and clicks on the "Sign up as a fanpage admin" button 2. The systems creates a new account. 3. The systems invokes the log in function to log the user into his account 4. The system forwards the user to his homepage for an fanpage admin user.
Alternative flow:	If the user enters invalid information or not enough information the system will redirect him to the log in page to fill or correct the missing or wrong information.

Use case: Log-out

Table 14: Use case: Log-out

Description:	If the user leaves the computer he needs to log himself out.
Actors:	• User • Page admin • Admin • User Management Service
Pre-Conditions:	• User is logged-in.
Post-Conditions:	**Success guarantees:** • The user is logged out of the system and needs to reenter his credential before using the service again.
Normal flow:	1. The user selects the ”Log-out” item from the drop down menu on the top of the page. 2. The system refreshes the page. 3. The system presents the Log-in page, showing that the user was successfully logged out.

Use case: Switch to connected user

Table 15: Use case: Switch to connected user

Description:	If there are fan page administrator profiles or system administrator profiles connected to a user it is possible to quick switch between them.
Actors:	• User • Page admin • Admin • User Management Service
Pre-Conditions:	• Normal User, Fanpage admin or System administrator is logged-in. • There are other profiles connected to the current user
Post-Conditions:	**Success guarantees:** • The user is able to use the service, as if he would have logged himself in as the connected user.
Normal flow:	1. The user selects the name of the connected profile from the drop down menu on the top of the page. 2. The system refreshes the page. 3. The system presents the homepage of the selected profile.

3.3.3 Postings use-case

The Posting use-case describes how a user or fanpage can share information with other users by posting and commenting. The UML use case diagram for this use case can be found in appendix A.3 on page 139.

Use case: Publish posts

Table 16: Use case: Publish posts

Description:	If a user of fanpage want to share information with their friends or the whole network they can publish a post.
Actors:	• User • Page Admin • Posting and Comment Service
Pre-Conditions:	• The user is logged into the system. • The user is viewing his homepage.
Post-Conditions:	**Success guarantees:** • The system stores the status post and presents it to the people who are allowed to view the post.

Continued on next page

Table 16 – continued from previous page

Normal flow:	1. The user enters the status he wants to publish in the "Update your status here" textfield. 2. The user presses the "Post" button to publish the post. 3. The system checks the input and stores the post. 4. The post appears in the profile of the user.
Alternative flow:	If the user is a normal (premium) user he is able to mark a post as *private*, so it can only be viewed by friends. If the status was empty the system will give an error feedback.

Use case: Comment on posts

Table 17: Use case: Comment on posts

Description:	If a user wants to react to a status he is able to comment on the post.
Actors:	• User • Page Admin • Friend / Page Following Management Service • Posting and Comment Service

Continued on next page

Table 17 – continued from previous page

Pre-Conditions:	• The user is logged into the system. • The user is viewing the status he wants to comment on.
Post-Conditions:	**Success guarantees:** • The system stores the comment to the post and notifies every user who is following the post.
Normal flow:	1. The user presses the "Comment" button on the post. 2. On the comment page he needs to fill the "Post your comment here" form. 3. The user needs to press the "Comment" button to submit his comment. 4. The Friend / Page Following Management Service is checking if the user is allowed to comment on the post. 5. The system stores the comment and notifies all other user who are following the post. 6. The system refreshes the page and the comment appears.
Alternative flow:	If the user is not allowed to comment on the status, the comment will not be accepted and the system will give an error feedback. If the comment was empty the system will give an error feedback.

Use case: Read posts

Table 18: Use case: Read posts

Description:	To keep track of friends or fanpages the user can read the shared posts of friends and pages.
Actors:	- User - Page Admin - Friend / Page Following Management Service - Posting and Comment Service
Pre-Conditions:	- The user is logged into the system. - The user is allowed to read the post.
Post-Conditions:	**Success guarantees:** - The comment is displayed on the users screen.
Normal flow:	1. The user selects the user/fanpage he wants to read the post. 2. The system checks which posts the user is allowed to view. 3. The system loads the appropriate posts. 4. The posts are displayed on the page.
Alternative flow:	If the user is not allowed to view a post, the post is not going to be displayed. In the worst case the system is not showing any post.

Use case: View Notifications

Table 19: Use case: View Notifications

Description:	If a user published a post or comment on a post he gets a notification if there are new comments on the post.
Actors:	• User • Page Admin • Posting and Comment Service
Pre-Conditions:	• The user is logged into the system. • The user published a post or comment on a post. • Another user commented on the followed post.
Post-Conditions:	**Success guarantees:** • The user is able to see the comment he was notified about.
Normal flow:	1. The user selects the "Notifications" entry on the top. 2. The systems collects all posts with new notifications. 3. The system lists all followed posts with new comments.
Alternative flow:	If there are no new notifications the "Notifications" page is empty.

Use case: Up/Down vote posts/comments

Table 20: Use case: Up/Down vote Posts/Comments

Description:	To show if a user agrees or disagrees with the opinion of another user, he is able to up or down vote the comment.
Actors:	• User • Page Admin • Friend / Page Following Management Service • Posting and Comment Service
Pre-Conditions:	• The user is logged into the system. • The user is allowed to read the post/comment.
Post-Conditions:	**Success guarantees:** • The ”Karma” rating is changed.

Continued on next page

Table 20 – continued from previous page

Normal flow:	1. The user selects the comment/post he wants to vote. 2. The user either up or down votes the post/comment. 3. The system checks if the user is allowed to vote the comment. 4. The system stores the vote. 5. The page is refreshed and the karma rating is changed according to the vote.
Alternative flow:	If the user is not allowed to vote the post/comment, the karma of the post/comment is not going to be changed.

Use case: Edit posts/comments

Table 21: Use case: Edit posts/comments

Description:	If a user wants to change a previous post or comment he is able to edit his post/comment.
Actors:	• User • Page Admin • Friend / Page Following Management Service • Posting and Comment Service

Continued on next page

Table 21 – continued from previous page

Pre-Conditions:	• The user is logged into the system. • The user has written a comment/post.
Post-Conditions:	**Success guarantees:** • The comment/post is changed.
Normal flow:	1. The user selects the comment / post he wants to edit and clicks on the ”Edit” button. 2. The system provides a text box where the user can edit his comment / post. 3. The user changes his comment / post and submits the change. 4. The system checks if the user is allowed to change this comment / post. 5. The system updates all relevant information. 6. The page is getting refreshed and the comment / post is changed.
Alternative flow:	If the user is not allowed to change the comment/post or deletes the complete text within the comment/post, the comment/post is not going to be changed.

Use case: Delete posts/comments

Table 22: Use case: Edit posts/comments

Description:	A user can delete a previous post or comment.

Continued on next page

Table 22 – continued from previous page

Actors:	- User - Page Admin - Friend / Page Following Management Service - Posting and Comment Service
Pre-Conditions:	- The user is logged into the system. - The user has written a comment/post.
Post-Conditions:	**Success guarantees:** - The comment/post is deleted.
Normal flow:	1. The user selects the comment/post he wants to delete and clicks on the ”Delete” button. 2. The system checks if the user is allowed to delete this comment/post. 3. The system updates deletes the comment/post. 4. The page is getting refreshed and the comment/post is deleted.
Alternative flow:	If the user is not allowed to delete the comment/post, the comment/post is not going to be deleted.

Use case: View latest posts

Table 23: Use case: View latest posts

Description:	On the homepage of the user the latest posts of his friends and followed pages are presented. As a fanpage all posts of the fanpage itself are presented.
Actors:	• User • Page Admin • Friend / Page Following Management Service • Posting and Comment Service
Pre-Conditions:	• The user is logged into the system.
Post-Conditions:	**Success guarantees:** • The latest posts of friends and followed pages are presented.
Normal flow:	1. The user returns to his homepage by clicking on the "The Network" button on the top of the page or logs himself into the network. 2. The system gathers the latest posts of the friends and followed pages. 3. The latest posts of friends and followed pages are presented.
Alternative flow:	If there are not post which could be presented, the page is going to be empty.

3.3.4 Manage profile/page use-case

For a social network user it is important to keep his profile page or fanpage up-to-date. This use case describes the way to update and manage the profile/page. The UML use case diagram for this use case can be found in appendix A.4 on page 140.

Use case: Add/Update personal information

Table 24: Use case: Add/Update personal information

Description:	A normal user needs to change their personal information.
Actors:	- User - User Management Service
Pre-Conditions:	- The user is logged into the system. - The user opened the profile editing page.
Post-Conditions:	**Success guarantees:** - The personal information of the user are updated.

Continued on next page

Table 24 – continued from previous page

Normal flow:	1. The user changes/adds the information he wants. 2. The user submits the change by clicking on the ”Submit” button. 3. The system updates the profile information of the user. 4. The page is getting refreshed and the updated information are presented.
Alternative flow:	If the user is leaving a mandatory field blank the informations are not going to get updated. If the user presses the ”Discard” button no information are updated.

Use case: Add/Update page information

Table 25: Use case: Add/Update page information

Description:	A page admin needs to change their page information.
Actors:	• Page Admin • User Management Service
Pre-Conditions:	• The page admin is logged into the system. • The page admin has opened the fanpage editing page.

Continued on next page

Table 25 – continued from previous page

Post-Conditions:	**Success guarantees:** • The fanpage information of the fanpage are updated.
Normal flow:	1. The fanpage admin changes/adds the information he wants. 2. The fanpage admin submits the change by clicking on the "Submit" button. 3. The system updates the fanpage information. 4. The page is getting refreshed and the updated information are presented.
Alternative flow:	If the fanpage admin is leaving a mandatory field blank the informations are not going to get updated. If the fanpage admin presses the "Discard" button no information are updated.

Use case: Add/Update profile/page picture

Table 26: Use case: Add/Update profile/page picture

Description:	A page admin or user needs to update their profile / page picture.
Actors:	• User • Page Admin • User Management Service

Continued on next page

Table 26 – continued from previous page

Pre-Conditions:	- The user/page admin is logged into the system. - The user/page admin has opened the fanpage/profile editing page.
Post-Conditions:	**Success guarantees:** - The picture is updated.
Normal flow:	1. The user/fanpage admin clicks the ”Change picture” button. 2. The user/fanpage is asked to select a picture from his computer. 3. The systems replaces the old picture with the new one. 4. The page gets refreshed presenting the new picture.
Alternative flow:	If the user is not selecting any new picture, a picture which extends the allowed size or uses an unknown format, the picture is not updated.

Use case: Change E-Mail address

Table 27: Use case: Change E-Mail address

Description:	If f page admin or user is changing his e-mail address or looses access to his e-mail account he needs to change it in the social network as well.

Continued on next page

Table 27 – continued from previous page

Actors:	• User • Page Admin • User Management Service
Pre-Conditions:	• The user/page admin is logged into the system. • The user/page admin has opened the fanpage/profile editing page.
Post-Conditions:	**Success guarantees:** • The e-mail address is updated.
Normal flow:	1. The user/fanpage admin enters his new e-mail address. 2. The user/fanpage admin clicks the "Save" button. 3. The system checks if the input is valid. 4. The system saves the new e-mail address. 5. The page gets refreshed presenting the new e-mail address.
Alternative flow:	If the user is not entering a valid e-mail address, it is not updated.

Use case: Change password

Table 28: Use case: Change password

Description:	Because of security reasons it is recommend to change the password regularly.
Actors:	- User - Page Admin - User Management Service
Pre-Conditions:	- The user/page admin is logged into the system. - The user/page admin opened the fanpage/profile editing page.
Post-Conditions:	**Success guarantees:** - The password is updated.

Continued on next page

Table 28 – continued from previous page

Normal flow:	1. The user/fanpage admin enters his current password in the ”Old Password” textfield. 2. The user/fanpage admin enters his new password in the ”New Password” and ”Retype Password” textfield. 3. The user/fanpage admin clicks the ”Save” button. 4. The system validates the old password and replaces it with the new password. 5. The user/fanpage admin gets notified that the password was updated.
Alternative flow:	If the user/fanpage admin entered a wrong old password or the ”New Password” and ”Retype Password” fields are not identical the system will not update the password and notify the user.

Use case: Remove connected page/admin

Table 29: Use case: Remove connected page/admin

Description:	Every user can be a page admin/system admin. He –and only he– can delete this connection at any time.
Actors:	• User • User Management Service

Continued on next page

Table 29 – continued from previous page

Pre-Conditions:	• The user is logged into the system. • The user has a connection with a fanpage and/or system admin account. • The user opened the profile editing page.
Post-Conditions:	**Success guarantees:** • The connection between the user types is deleted.
Normal flow:	1. The user selects the connection he wants to delete. 2. The user clicks on the ”Remove” button next to the connection. 3. The system removes the connection. 4. The page gets refreshed showing that there is no longer a connection.

Use case: Delete account

Table 30: Use case: Delete account

Description:	It could be possible that user/fanpage admin don't want to use the service anymore and delete their profile.

Continued on next page

Table 30 – continued from previous page

Actors:	• User • Page Admin • System Administrator • User Management Service
Pre-Conditions:	• The user/page admin/system administrator is logged into the system. • The user/page admin opened the fanpage/profile editing page. (System administrator found the user within the ”Manage User” section).
Post-Conditions:	**Success guarantees:** • The user and his personal profile is deleted.
Normal flow:	1. The user/fanpage admin/system administrator presses the ”Delete user”/”Delete fanpage” button. 2. The system asks for confirmation. 3. The system is deleting the profile of the user and performs a logout of the user.

Use case: Reset password

Table 31: Use case: Reset password

Description:	If the user looses the credentials for his account the administrator is able to reset the password.
Actors:	• System Administrator • User Management Service
Pre-Conditions:	• The system administrator is logged into the system. • System administrator found the user within the ”Manage User” section.
Post-Conditions:	**Success guarantees:** • The password is reseted and the user gets an e-mail with his temporary password.
Normal flow:	1. The system administrator presses the ”Reset password” button. 2. The system resets the password to a temporary password. 3. The user receives the temporary password via e-mail.

Use case: Add connected user

Table 32: Use case: Add connected user

Description:	Every system administrator and fanpage administrator can connect with one normal user of the network to have quick access to all functionalities from his normal profile.
Actors:	• Fanpage Administrator • System Administrator • User Management Service
Pre-Conditions:	• The fanpage/system administrator is logged into the system.
Post-Conditions:	**Success guarantees:** • The normal user is connected to the profile.

Continued on next page

Table 32 – continued from previous page

Normal flow:	**Fanpage administrator:** 1. The fanpage administrator enters the e-mail address or the user id of the user he wants to connect with in the "Enter email or User ID" textfield on the fanpage editing page. 2. The fanpage administrator presses the "Connect" button. 3. This connects the user with the fanpage. **System administrator:** 1. The system administrator finds the user within the "Manage user" section. 2. The fanpage administrator presses the "Connect with user" button next to the profile. 3. The system connects the user with the fanpage.
Alternative flow:	If the entered user id or email could not be associated to any user the user is not connected and the system will return an error notification.

3.3.5 Messages

One of the main features of a social network is the possibility writing messages to your friends. This use case describes the way to write and receive messages. The UML use case diagram for this use case can be found in appendix A.5 on page 141.

Use case: Start a new conversation

Table 33: Use case: Start a new conversation

Description:	A user (A) wants to text a person (B).

Continued on next page

Table 33 – continued from previous page

Actors:	• User • Friend / Page Following Management Service • Message Service
Pre-Conditions:	• The user (A) is logged into the system. • The user (A) is a friend of the user (B) he wants to text.
Post-Conditions:	**Success guarantees:** • The user (B) receives the message of the user (A).
Normal flow:	1. The user (A) accesses the "Message" section via the "Messages" button on the top of the page. 2. The user (A) searches for the user (B) using the search box or selects him from the recent messages listed. 3. The user (A) enters a message. And presses the "Send message" button. 4. The system receives the message and notifies the user (B).

Continued on next page

Table 33 – continued from previous page

Alternative flow:	To start the conversation the user (A) can also access the profile of the user (B) and press the "Send message" button. If the entered message is too long or empty the system will reject the message and return an error to the user.

Use case: Check all conversations

Table 34: Use case: Check all conversations

Description:	A user checks all of his conversations, to see if there are any new messages.
Actors:	• User • Message Service
Pre-Conditions:	• The user (A) is logged into the system.
Post-Conditions:	**Success guarantees:** • The user sees a list of all conversations.
Normal flow:	1. The user accesses the "Message" section via the "Messages" button on the top of the page. 2. The system collects all conversations of the user. 3. All conversations are presented to the user on the "Messages" page.

Use case: View conversation

Table 35: Use case: View conversation

Description:	A user (A) wants to read a conversation between him and another person (B).
Actors:	• User • Message Service • Friend / Page Following Management Service
Pre-Conditions:	• The user (A) is logged into the system. • The user (A) had a conversation with user (B) earlier.
Post-Conditions:	**Success guarantees:** • The user sees the conversation between him and the other user (B)
Normal flow:	1. The user (A) accesses the ”Message” section via the ”Messages” button on the top of the page. 2. The user (A) searches for the user (B) using the search box or selects him from the recent messages listed. 3. The system loads the conversation between the users. 4. The conversation is presented to the user (A).

Continued on next page

Table 35 – continued from previous page

Alternative flow:	To see the conversation the user (A) can also access the profile of the user (B) and press the "Send message" button. If there is no conversation between the users the system will show an empty conversation.

Use case: Reply to a message

Table 36: Use case: Reply to a message

Description:	A user (A) receives a message from person (B) and wants to reply to this message.
Actors:	• User • Message Service
Pre-Conditions:	• The user (A) is logged into the system. • The user (B) sent a text to user (A).
Post-Conditions:	**Success guarantees:** • The user (B) receives the reply of the user (A).

Continued on next page

Table 36 – continued from previous page

Normal flow:	1. The user (A) accesses the "Message" section via the "Messages" button on the top of the page. 2. The user (A) searches for the unread message of user (B) presented on the "Message" page and enters the conversation. 3. The user (A) enters a message. And presses the "Send message" button. 4. The system receives the message and notifies the user (B).
Alternative flow:	If the entered message is too long or empty the system will reject the message and return an error to the user.

3.4 Web application design

To create an intuitive service for the end user, the most important thing is to create an easy to use and clear design.

3.4.1 Page navigation diagram

The page navigation diagram can be found in appendix B on page 142. In the following passage the structure of the page map is going to be described.

The page map consists of 4 main areas: The public part, on top of the diagram, can be accessed by every user of the internet. If the user is subscribed to *The Network* he has the possibility to log in using his credentials. Otherwise every user can create a free account to the network. After logging in the user is able to access the area fitting to his user type. The left part is for fanpage administrators only, the middle part is for every normal user and the right part is getting accessed by system administrators. If the user profile is connected to any administrator account it is possible to switch between the areas quickly without the need of providing the credentials every time.

3.4.2 User Interface

To get the feeling how the social network is going to look like there are static wireframes available in appendix C on page 143. These wireframes show nearly every page of the service. To see the connection between each frame it is useful to check the page map in appendix B on page 142. Supplementing, the background of every frame is coloured depending on the usertype accessing the frame.

To provide a clean and simple UI the *Twitter Bootstrap* framework is going to be used. That is a very simple and popular tool to design webpages. It offers predefined HTML and CSS files which can be included in the web project to provide a clean and consistent design. There exist numerous themes for the framework, so it is going to be easy to find one, which is going to fit well to the project. Additional it is licensed under the *Apache License 2.0* which allows the free use and modification of the source code.

3.5 Design Patterns

Within the development of software products programmer are facing several recurring problems. Hence they started developing templates to solve them. These templates are called design patterns. Within the project there are going to be challenges solvable by these patterns. In the following sections the templates which are going to be used are getting specified.

3.5.1 Model-View-Controller

The model view controller is a template to structure a software product. The controller consists of three parts: the controller, the model and the view. A general component diagram of the model view controller used in this project can be found in appendix E on page 145. By using this pattern it is possible to separate the business logic from the view. On top of that it is possible to manage access control fairly easy within the controller.

3.5.2 Behavioral Design Patterns: *Command Pattern*

To realize the model-view controller it is important for the controller to easily execute commands within the business logic. To solve this problem there is a behavioral design pattern –the command pattern– to manage this execution. This command pattern is part of the *Gang of four*-templates [Gam+94]. The command pattern is going to be used within the project to simplify the manipulation and gathering of data.

3.5.3 Creational Design Patterns: *Factory Method*

To create the commands there is the need of a creation method. This problem can be resolved using a creational design pattern. The factory method is such a pattern simplifying the generation of an object. Within this pattern a class has a static function returning a new object. This pattern is part of the *Gang of four*-templates [Gam+94]. The factory pattern is going to be used within the project to simplify the construction of objects.

3.5.4 Architectural Pattern: *Active Record Pattern*

Since all data is going to be stored in a database the effective gathering of the information is a huge problem. But there are several design patterns solving this problem. For example the active record pattern simplifies the database communication. This pattern was specified by Fowler [Fow03]. The pattern represents every data row as an active record class. This class offers static functions to query the database and functions to update, delete and insert data. The active record pattern is going to be used within the project as a design pattern for the database communication.

3.6 Database

One of the most important parts of a social network is a powerful database, managing all accruing data. All the data needs to be stored persistent, while still being easily accessible. The social network is going to use a relational database. Within this project the build-in *JavaDB* (Apache Derby Database) is used to proof functionality. It needs to be considered that this database needs to replaced by an enterprise solution eg. *Oracle DB*, if the amount of users is increasing. This step is needed to provide a sustainable service even is the amount of data is rapidly incrementing.

3.6.1 Conceptual Schema: ER-Diagram

To create a durable database it is important to have a precise plan of the design. The first step –the conceptual schema of a relational database– can be expressed as an entity-relationship model (ER-model). The specific ER-model for *The Network* can be found in appendix D on page 144. In the following section the ER-model for *The Network* is going to be described.

Every user type is described by a table ('*User*', '*Admin*', '*Fanpage*'). Every entity of these tables is a part of the social profile of each user. To provide the connection between the user types there exists a foreign key connection between the tables ('*Admin* may be *User*', '*User* administrates *Fanpage*'). The connection between users and other users and users and fanpages is of cardinality [n:m], that's why there are going to be tables describing these connections ('*User* is friend with *User*', '*User* follows *Fanpage*'). The same thing applies for messages ('*Messages*') and status posts ('*Post*') ('*User* receives *Message*', '*User* sends *message*', '*User* publishes *Post*', '*Fanpage* publishes *Post*'). On top of that it is obvious that every post can have multiple comments ('*Comments*'), concluding there is going to be a [1:n] connection ('*Post* has *Comments*'). Every comment is published by a user ('*User* publishes *Comments*', '*Fanpage* publishes *Comments*'). To keep track of notifications for new comments on posts it is important to know if there has been any change since the last check of the user. This functionality is achieved by introducing a [n:m] connection between users and the post they published or commented on. By checking this connection ('*User* follows *Post*', '*Fanpage* follows *Post*') for a read flag it is possible to know if there are new notifications for the user.

3.6.2 Relational Schema according to Kemper/Eickler

For an easy implementation of the database the ER-model is not the most handy thing. That is why a relational schema is going to be derived out of the ER-model. This scheme is going to be described according to Kemper and Eickler [KE06].

Main tables

Listing 1: *User*-table

```
User: {[
    UserID: String,
    DisplayName: String,
    FirstName: String,
    LastName: String,
    DateOfBirth: Date,
    RelationshipStatus: String,
    Gender: String,
    Email: String,
    Street: String,
    HouseNr: Integer,
    Town: String,
    Zip: String,
    Picture: String,
    Premium: Boolean,
    Password: String
]}
```

Listing 2: *Admin*-table

```
Admin: {[
    AdminID: String,
    Email: String,
    Password: String,
    ConnectedUser → User
]}
```

Listing 3: *Fanpage*-table

```
Fanpage: {[
    PageID: String,
    PageName: String,
    DisplayName: String,
    Subject: String,
    Email: String,
    Password: String,
    Picture: String,
    Premium: Boolean,
    AdministratingUser → User
]}
```

Secondary tables

Listing 4: *Messages*-table

```
Message: {[
    MessageID: String,
    Timestamp: Date,
    Content: String,
    Read: Boolean,
    WritingUser → User,
    ReceivingUser → User
]}
```

Listing 5: *Post*-table

```
Post: {[
    PostID: String,
    Timestamp: Date,
    Public: Boolean,
    Karma: Integer,
    Content: String,
    PublishingUser → User,
    PublishingPage → Fanpage
]}
```

Listing 6: *Comments*-table

```
Comment: {[
    CommentID: String,
    Timestamp: Date,
    Karma: Integer,
    Content: String,
    RelatedPost → Post
    PublishingUser → User,
    PublishingPage → Fanpage
]}
```

Connection tables

Listing 7: *User is friend with user*-table

```
UisFriendWithU: {[
    User1 → User,
    User2 → User,
    Accepted: Boolean,
    Notified: Boolean
]}
```

Listing 8: *User follows fanpage*-table

```
UfollowsF: {[
    FollowingUser → User,
    FollowedFanpage → Fanpage,
]}
```

Listing 9: *User follows post*-table

```
UfollowsP: {[
    FollowingUser → User,
    FollowedPost → Post,
    Read: Boolean
]}
```

Listing 10: *Fanpage follows post*-table

```
FfollowsP: {[
    FollowingFanpage → User,
    FollowedPost → Post,
    Read: Boolean
]}
```

In total there are ten tables, all connected with each other. The cardinality of several connections is [1:n].

3.6.3 Normalisation

It is important to normalize all tables within a database to prevent redundancy and ensure atomic data structures. The design of the database shown in the conceptual schema (Appendix D on page 144) as well as in the relational schema (Chapter 3.6.2 on page 57) are both normalized according to the 3rd normal form. That is done, because the 3rd normal form is the best tradeoff between non-redundancy architecture and performance.

3.7 Component diagrams

The following section will show how user transactions are going to be handled by the system in detail, using component diagrams. Because the user transactions are all more or less similar only eight of them are going to be described in detail so this section is not going to be overloaded with unnecessary information.

3.7.1 Query the database: *Get all friends*

This user transaction is characterized by the use case described in table 3 on page 14. The component diagram for this user transaction is shown in appendix E.2 on page 147. At the current planing status the following SQL statement would give all rows of friends. Note: these queries are not optimized and may change during development. Moreover these statements were not tested, since there is no running system. This means there could be errors within the command or the command could be incomplete, nevertheless they can be seen as pseudo code. The *UserIDOfCurrentUser* is depending on the current user.

Listing 11: *Get all friends* SQL statement

```sql
Select User.UserID, User.DisplayName
From UisFriendWithU
Inner Join User
On UisFriendWithU.User2.UserID = User.UserID
```

```
Where UisFriendWithU.User1.UserID =
    UserIDOfCurrentUser ;
```

3.7.2 Query the database: *Get all new notifications*

This user transaction is characterized by the use case described in table 19 on page 32. The component diagram for this user transaction is shown in appendix E.3 on page 148. At the current planing status the following SQL statement would give all rows of post with new comments. Note: these queries are not optimized and may change during development. Moreover these statements were not tested, since there is no running system. This means there could be errors within the command or the command could be incomplete, nevertheless they can be seen as pseudo code. The *UserIDOfCurrentUser* is depending on the current user.

Listing 12: *Get all new notifications* SQL statement

```
Select Post.PostID, Post.Content
From UFollowsP
Inner Join Post
On Post.PostID = UFOllowsP.FollowedPost.PostID
Where UFollowsP.FollowingUser.UserID =
    UserIDOfCurrentUser And UFollowsP.Read = 0;
```

3.7.3 Query the database: *Get all conversation*

This user transaction is characterized by the use case described in table 34 on page 51. The component diagram for this user transaction is shown in appendix E.4 on page 149. At the current planing status the following SQL statement would give all rows of post with new comments. Note: these queries are not optimized and may change during development. Moreover these statements were not tested, since there is no running system. This means there could be errors within the command or the command could be incomplete, nevertheless they can be seen as pseudo code. The *UserIDOfCurrentUser* is depending on the current user.

Listing 13: *Get all conversation* SQL statement

```
Select ReceivingUser.UserID
```

```
From Messages
Where WritingUser.UserID = UserIDOfCurrentUser;

 Select WritingUser.UserID
From Messages
Where ReceivingUser.UserID = UserIDOfCurrentUser;
```

3.7.4 Query the database: *Get all posts of friends and followed fanpages*

This user transaction is characterized by the use case described in table 23 on page 37. The component diagram for this user transaction is shown in appendix E.5 on page 150. At the current planing status the following SQL statement would give all rows of post published by friends or followed pages. Note: these queries are not optimized and may change during development. Moreover these statements were not tested, since there is no running system. This means there could be errors within the command or the command could be incomplete, nevertheless they can be seen as pseudo code. The *UserIDOfCurrentUser* is depending on the current user.

Listing 14: *Get all posts of friends and followed fanpages* SQL statement

```
 Select Post.PostID, Post.Content
From UisFriendWithU
 Inner Join Post
On UisFriendWithU.User2.UserID = Post.PublishingUser.
    UserID
Where UisFriendWithU.User1.UserID =
    UserIDOfCurrentUser;

 Select Post.PostID, Post.Content
From UfollowsF
 Inner Join Post
On UfollowsF.FollowedFanpage.PageID = Post.
    PublishingPage.PageID
Where UfollowsF.FollowingUser.UserID =
    UserIDOfCurrentUser;
```

3.7.5 Insert data in the database: ***Create user***

This user transaction is characterized by the use case described in table 12 on page 24. The component diagram for this user transaction is shown in appendix E.6 on page 151. At the current planing status the following SQL statement would insert all data provided by the new user as a row within the user table. Note: these queries are not optimized and may change during development. Moreover these statements were not tested, since there is no running system. This means there could be errors within the command or the command could be incomplete, nevertheless they can be seen as pseudo code. *UserInput_*[...] is depending on the user's input.

Listing 15: *Create user* SQL statement

```
Insert Into User(UserID, DisplayName, FirstName,
    LastName, DateOfBirth, RelationshipStatus, Gender,
    EMail, Street, HouseNr, Town, Zip, Picture, Premium
    , Password)
Values (NextUserID, UserInput_DisplayName,
    UserInput_FirstName, UserInput_LastName,
    UserInput_DateOfBirth, UserInput_RelationshipStatus
    , UserInput_Gender, UserInput_EMail,
    UserInput_Street, UserInput_HouseNr, UserInput_Town
    , UserInput_Zip, null, false, UserInput_Password);
```

3.7.6 Insert data in the database: ***Publish post***

This user transaction is characterized by the use case described in table 16 on page 28. The component diagram for this user transaction is shown in appendix E.7 on page 152. At the current planing status the following SQL statement would insert all data provided by the user as a row within the post table. Note: these queries are not optimized and may change during development. Moreover these statements were not tested, since there is no running system. This means there could be errors within the command or the command could be incomplete, nevertheless they can be seen as pseudo code. *UserInput_*[...] is depending on the user's input, *CurrentUser* is depending on the user publishing the post.

Listing 16: *Publish Post* SQL statement

```
Insert Into Post(PostID, Timestamp, Public, Karma,
    Content, PublishingUse, PublishingPage)
Values (NextPostID, CurrentTime, UserInput_Public, 0,
    UserInput_Content, CurrentUser, null);
```

3.7.7 Delete data from the database: *Delete post*

This user transaction is characterized by the use case described in table 22 on page 35. The component diagram for this user transaction is shown in appendix E.8 on page 153. At the current planing status the following SQL statement would delete the row containing the post and all comments connected to the post. Note: these queries are not optimized and may change during development. Moreover these statements were not tested, since there is no running system. This means there could be errors within the command or the command could be incomplete, nevertheless they can be seen as pseudo code. *CurrentPost* is depending on the post which needs to be deleted.

Listing 17: *Delete Post* SQL statement

```
Delete From Post
Where PostID = CurrentPost;

Delete From Comment
Where RelatedPost.PostID = CurrentPost;
```

3.7.8 Update data in the database: *Update profile information*

This user transaction is characterized by the use case described in table 24 on page 38. The component diagram for this user transaction is shown in appendix E.9 on page 154. At the current planing status the following SQL statement would update the row containing the profile information of the user. Note: these queries are not optimized and may change during development. Moreover these statements were not tested, since there is no running system. This means there could be errors within the command or the command could be incomplete, nevertheless they can be seen as pseudo code.

*UserInput_*[...] is depending on the user's input and *UserIDOfCurrentUser* is depending on the current user.

Listing 18: *Update Profile Information* SQL statement

```sql
Update user
Set DisplayName = UserInput_DisplayName , FirstName =
    UserInput_FirstName , LastName = UserInput_LastName ,
     DateOfBirth = UserInput_DateOfBirth ,
    RelationshipStatus = UserInput_RelationshipStatus ,
    Gender = UserInput_Gender , EMail = UserInput_EMail ,
     Street = UserInput_Street , HouseNr =
    UserInput_HouseNr , Town = UserInput_Town , Zip =
    UserInput_Zip , Picture = UserInput_Picture ,
    Password =  UserInput_Password
Where UserID = UserIDOfCurrentUser
```

3.8 Technical Product Environment

While development and deployment it is very important to know which is going to be the technical environment of the service.

3.8.1 Development Environment

The development is going to be done on a single laptop. To secure the data regular backups to a hard disc drive are scheduled. On top of that there is an instant synchronization to a cloud environment, where the data is stored to be accessible everywhere. The development environment is going to have the specification defined in table 37 on the following page.

3.8.2 Server Environment

After the development the service needs to be deployed to a server for testing first and for 24/7 operations later. For the testing timeframe a server with the specifications in table 38 on the next page are sufficient. For a reliable full time operation better hardware specifications are recommended.

Hardware:	
Model	MacBook Pro Mid 2010
Software:	
Operating system	Mac OS X 10.9.2 (x86_64)

Runtime environment	Java version 1.7.0 Update 25;
	Java(TM) SE Runtime Environment(build 1.7.0_25-b15);
	Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)

Integrated development environment	Netbeans 7.4
Application server	GlassFish Server Open Source Edition 4.0 (build 89)
Database	Apache Derby Database 10.9.1.0

Table 37: Specification of the development environment

Hardware:	
Model	Virtual Server
Software:	
Operating system	Ubuntu 112_04 LTS (amd64)
Runtime environment	Java version 1.7.0_51 OpenJDK Runtime Environment (IcedTea 2.4.4) (7u51-2.4.4-0ubuntu0.12.04.2) OpenJDK 64-Bit Server VM (build 24.45-b08, mixed mode)
Web server	Apache/2.2.22 (Ubuntu)
Application server	GlassFish Server Open Source Edition 4.0 (build 89)
Database	Apache Derby Database 10.9.1.0

Table 38: Specification of the development environment

3.8.3 Client Environment

To use the social network there are several minimum specification the client's device should fulfill to enjoy the service without any problems. These specification can be found in table 39. The runtime environment is only needed for the Java client, which is an optional requirement.

Software:	
Operating system	Windows 7 (or newer) Ubuntu 12.10 (or compatible) Mac OS X 10.8 iOS 7 (or newer) Android 4.3 (or newer)
Web Browser	Internet Explorer 8 (or newer) Firefox 24 (or newer) Web browser with latest WebKit (Safari 6, Chrome 30, or newer)
Runtime environment	Java Runtime Environment Version 7 Update 51 (or newer)

Table 39: Specification of the client's environment

3.9 Risk assessment

For the successful realization of a project it is important to minimize the risks, which may lead to a failure of the project, proactive. To prevent a risk it is important to know the most important ones. These risk analysis is done within the risk assessment table, which can be found in appendix F on page 155. If a risk has an high impact and an high probability it needs to be monitored, so it can be avoided. The only risk within this project which has both –high impact and high probability– is the risk of insufficient time. To prevent this risk it is highly important to stick to the plan and invest enough time into the project.

3.10 Test Plan

Within the waterfall model the test plan is created while development or after the development. There are several problems with this approach, for example the test cases could not meet the requirements, or unnecessary work was done while development. Because of that *Test driven development* (TDD) was introduced. Within TDD the test cases get developed before the programming is started. This makes sure that only the necessary code is written and all requirements are met. Within this section all test cases get developed and described. Most of the test cases refer to use-cases which were developed earlier. All tests are going to be black-box tests. For the actual testing the additional column ”Actual result” needs to be added and later filled by the tester.

3.10.1 Access control

This section of test cases is about accessing the network as one of the user types.

Login as normal user

Table 40: Test case: Login as normal user

Test case name: Login as normal user		
This test case refers to the use case described in table 11 on page 22. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Open the URL of "The Network".	The login page is displayed.
2	Enter the credentials of a normal user.	
3	Press the "Login" button.	The system starts loading a new page.
4	The page is completely loaded.	The system presents the home page of the normal user.

Login as fanpage admin

Table 41: Test case: Login as fanpage admin

Test case name: Login as fanpage admin		
This test case refers to the use case described in table 11 on page 22. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Open the URL of "The Network".	The login page is displayed.
2	Enter the credentials of a fanpage admin.	
3	Press the "Login" button.	The system starts loading a new page.

Continued on next page

Table 41 – continued from previous page

Step Nr.	Step Description	Expected result
4	The page is completely loaded.	The system presents the home page of the fanpage admin.

Login as system administrator

Table 42: Test case: Login as system administrator

Test case name: Login as system administrator		
This test case refers to the use case described in table 11 on page 22. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Open the URL of ”The Network”.	The login page is displayed.
2	Enter the credentials of a normal user.	
3	Press the ”Login” button.	The system starts loading a new page.
4	The page is completely loaded.	The system presents the home page of the normal user.

Login with invalid credentials

Table 43: Test case: Login with invalid credentials

Test case name: Login with invalid credentials		
This test case refers to the use case described in table 11 on page 22. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Open the URL of ”The Network”.	The login page is displayed.
2	Enter any invalid credentials.	

Continued on next page

Table 43 – continued from previous page

Step Nr.	Step Description	Expected result
3	Press the ”Login” button.	The system starts loading a new page.
4	The page is completely loaded.	The system presents an error message saying that the used credentials have been wrong.

Sign up as normal user

Table 44: Test case: Sign up as normal user

Test case name: Sign up as normal user		
This test case refers to the use case described in table 12 on page 24. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Open the URL of ”The Network”.	The login page is displayed.
2	Enter all demanded information underneath the ”Sign up as normal user” label using only valid data.	
3	Accept the General business terms.	
4	Press the ”Sign up as normal user” button.	The system start loading a new page.
5	The page is completely loaded.	The system presents the homepage of the new created user.

Signup as normal user using invalid input

Table 45: Test case: Sign up as normal user using invalid input

Test case name: Sign up as normal user using invalid input		
This test case refers to the use case described in table 12 on page 24. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Open the URL of ”The Network”.	The login page is displayed.
2	Enter all demanded information underneath the ”Sign up as normal user” label using invalid data some- or everywhere.	
3	Accept (or decline) the General business terms.	
4	Press the ”Sign up as normal user” button.	The system start loading a new page.
5	The page is completely loaded.	The system presents the login page, showing an error that the entered information have been invalid.

Signup as fanpage admin

Table 46: Test case: Sign up as fanpage admin

Test case name: Sign up as fanpage admin		
This test case refers to the use case described in table 13 on page 25. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Open the URL of ”The Network”.	The login page is displayed.

Continued on next page

Table 46 – continued from previous page

Step Nr.	Step Description	Expected result
2	Enter all demanded information underneath the ”Sign up as fanpage admin” label using only valid data.	
3	Accept the General business terms.	
4	Press the ”Sign up as fanpage admin” button.	The system start loading a new page.
5	The page is completely loaded.	The system presents the homepage of the new created fanpage.

Signup as fanpage admin using invalid input

Table 47: Test case: Sign up as fanpage admin using invalid input

Test case name: Sign up as fanpage admin using invalid input		
This test case refers to the use case described in table 13 on page 25. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Open the URL of ”The Network”.	The login page is displayed.
2	Enter all demanded information underneath the ”Sign up as fanpage admin” label using invalid data some- or everywhere.	
3	Accept (or decline) the General business terms.	
4	Press the ”Sign up as fanpage admin” button.	The system start loading a new page.

Continued on next page

Table 47 – continued from previous page

Step Nr.	Step Description	Expected result
5	The page is finished loading.	The system presents the log in page, showing an error that the entered information have been invalid.

Change user type according to connected user

Table 48: Test case: Change user type according to connected user

Test case name: Change user type according to connected user		
This test case refers to the use case described in table 15 on page 27. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select one of the connected profiles from the drop-down box on the top of the page.	The page gets refreshed.
2	The page is completely loaded.	The system presents the homepage of the connected profile.

Logout

Table 49: Test case: Logout

Test case name: Logout		
This test case refers to the use case described in table 14 on page 26. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Log out” entry from the drop-down box on the top of the page.	The page gets refreshed.
2	The page is completely loaded.	The system presents the login page.

3.10.2 Normal user

The following sections refers to all test cases using a normal user profile. All test cases require that a normal user is logged into the system.

View homepage

Table 50: Test case: View homepage

Test case name: View homepage		
This test case refers to the use case described in table 23 on page 37. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”The Network” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents all recent posts of friends or followed fanpages on the middle of the page. The header items have badges if there are new notifications, friend requests or messages.

Friend management

The following subsection refers to test cases using the friend management system of a normal user.

Show all friends

Table 51: Test case: Show all friends

Test case name: Show all friends		
This test case refers to the use case described in table 3 on page 14. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”Friends” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all not answered friend requests followed by a list of all friends.

Find user from the friend overview page

Table 52: Test case: Find user from the friend overview page

Test case name: Find user from the friend overview page		
Post-condition: User is on the friend overview page		
Step Nr.	Step Description	Expected result
1	Enter the name of the user you want to search in the search box on the top.	
2	Press enter.	A new page gets loaded.
3	The page is completely loaded.	The system presents a list of all users fitting to the search query.

Add a friend

Table 53: Test case: Add a friend

Test case name: Add a friend		
This test case refers to the use case described in table 6 on page 16. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press the "Add Friend" button on the profile of the user.	A new page gets loaded.
2	The page is completely loaded.	The "Add Friend" button is replaced by a "Request pending" button.

Accept a friend

Table 54: Test case: Accept a friend

Test case name: Accept a friend		
This test case refers to the use case described in table 7 on page 17. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the "Friends" button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all not answered friend requests followed by a list of all friends.
3	Press on the "Accept" button next to an open friend request.	The page gets refreshed.
4	The page is completely loaded.	The new added user is now part of the friends list.

Reject a friend

Table 55: Test case: Reject a friend

Test case name: Reject a friend		
This test case refers to the use case described in table 8 on page 19. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”Friends” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all not answered friend requests followed by a list of all friends.
3	Press on the ”Reject” button next to an open friend request from the top.	The page gets refreshed.
4	The page is completely loaded.	The rejected user is disappeared from the list.

Remove a friend

Table 56: Test case: Remove a friend

Test case name: Remove a friend		
This test case refers to the use case described in table 10 on page 21. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”Friends” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all not answered friend requests followed by a list of all friends.
3	Find the user which needs to get removed in the list.	

Continued on next page

Table 56 – continued from previous page

Step Nr.	Step Description	Expected result
4	Press on the ”Remove friend” button next to the name of the user.	The page gets refreshed.
5	The page is completely loaded.	The removed user is disappeared from the list.

Explore friend’s profile

Table 57: Test case: Explore friend’s profile

Test case name: Explore friend’s profile		
This test case refers to the use case described in table 18 on page 31. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”Friends” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all not answered friend requests followed by a list of all friends.
3	Find the user profile which needs to get explored.	
4	Press on the name of the user.	The page gets refreshed.
5	The page is completely loaded.	The profile of the user and all his posts are shown.

Fanpage management

The following subsection refers to test cases using the fanpage management system of a normal user.

Show all followed fanpages

Table 58: Test case: Show all followed fanpages

Test case name: Show all followed fanpages		
This test case refers to the use case described in table 4 on page 14. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”Fanpages” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all followed fanpages.

Find a fanpage from the fanpage overview page

Table 59: Test case: Find a fanpage from the fanpage overview page

Test case name: Find a fanpage from the fanpage overview page		
Post-condition: User is on the fanpage overview page		
Step Nr.	Step Description	Expected result
1	Enter the name of the fanpage you want to search in the search box on the top.	
2	Press enter.	A new page gets loaded.
3	The page is completely loaded.	The system presents a list of all fanpages fitting to the search query.

Follow a fanpage

Table 60: Test case: Follow a fanpage

Test case name: Follow a fanpage		
This test case refers to the use case described in table 5 on page 15. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press the ”Follow this page” button on the homepage of the fanpage.	A new page gets loaded.
2	The page is completely loaded.	The ”Follow this page” button is replaced by a ”Unfollow this page” button.

Unfollow a fanpage

Table 61: Test case: Unfollow a fanpage

Test case name: Unfollow a fannpage		
This test case refers to the use case described in table 9 on page 20. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”Fanpage” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of followed fanpages.
3	Find the fanpage you want stop to follow.	
4	Press on the ”Unfollow” button next to the name of the fanpage.	The page gets refreshed.
5	The page is completely loaded.	The un-followed page is disappeared from the list.

Explore fanpage

Table 62: Test case: Explore fanpage

Test case name: Explore fanpage		
This test case refers to the use case described in table 18 on page 31. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”Pages” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all followed fanpages.
3	Find the fanpage which needs to get explored.	
4	Press on the name of the fanpage.	The page gets refreshed.
5	The page is completely loaded.	The fanpage and all posts of it are shown.

Messages

The following subsection refers to test cases using the message system of a normal user.

View all conversations

Table 63: Test case: View all conversations

Test case name: View all conversations		
This test case refers to the use case described in table 34 on page 51. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press the ”Messages” button on top of the page.	The page gets reloaded.
2	The page is completely loaded.	The system presents a list of recent conversation.

Send message from profile

Table 64: Test case: Send message from profile

Test case name: Send message from profile		
This test case refers to the use case described in table 33 on page 49. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press the "Friends" button on top of the page.	The page gets reloaded.
2	The page is completely loaded	The system presents a list of all friends.
3	Enter the name of the user you want to send a messge to in the search box on the top.	
4	Press enter.	A new page gets loaded.
5	The page is completely loaded.	The system presents a list of all users fitting to the search query.
6	Press on the user name of the user you want to send a message to.	The page gets reloaded.
7	The page is completely loaded.	The system presents the profile page of the user.
8	Press on the "Send message" button.	The page gets reloaded.
9	The page is completely loaded.	The system presents the conversation page with the user (empty if there haven't been any messages to/from the user yet).
10	Enter the message in the provided text box and press "Send" button.	The page gets reloaded.

Continued on next page

Table 64 – continued from previous page

Step Nr.	Step Description	Expected result
11	The page is completely loaded.	The message appears in the conversation section of the page.

Send message from recent conversations

Table 65: Test case: Send message from recent conversations

Test case name: Send message from recent conversations		
This test case refers to the use case described in table 33 on page 49. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press the "Messages" button on top of the page.	The page gets reloaded.
2	The page is completely loaded.	The system presents a list of recent conversation.
3	Select a conversation from the list.	The page gets reloaded.
4	The page is completely loaded.	The system presents the conversation page with the user.
5	Enter the message in the provided text box and press "Send" button.	The page gets reloaded.
6	The page is completely loaded.	The message appears in the conversation section of the page.

Send message from ”Messages” page

Table 66: Test case: Send message from ”Messages” page

Test case name: Send message from ”Messages” page		
This test case refers to the use case described in table 33 on page 49. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press the ”Messages” button on top of the page.	The page gets reloaded.
2	The page is completely loaded.	The system presents a list of recent conversation.
3	Enter the name of the user you want to send a message to in the search box on the top of the page.	The page gets reloaded.
4	The page is completely loaded.	The system presents a list of user fitting to the search query.
5	Select one entry of the list by clicking on the name.	The page gets reloaded.
6	The page is completely loaded.	The system presents the conversation page with the user.
7	Enter the message in the provided text box and press ”Send” button.	The page gets reloaded.
8	The page is completely loaded.	The message appears in the conversation section of the page.

Send empty message

Table 67: Test case: Send empty message

Test case name: Send empty message		
This test case refers to the use case described in table 33 on page 49. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Enter a conversation, as described in table 66 on the preceding page, 65 on page 85 or 64 on page 84	The system presents the conversation page with the user.
2	Do not enter a message in the provided text box and press "Send" button.	The page gets reloaded.
3	The page is completely loaded.	The system provides an error and does not send the empty message.

Reply to message

Table 68: Test case: Reply to message

Test case name: Reply to message		
This test case refers to the use case described in table 36 on page 53. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press the "Messages" button on top of the page.	The page gets reloaded.
2	The page is completely loaded.	The system presents a list of recent conversation. If there are any new messages they are marked and placed on the top.
3	Select a marked entry in the list.	The page gets reloaded.

Continued on next page

Table 68 – continued from previous page

Step Nr.	Step Description	Expected result
4	The page is completely loaded.	The system presents the conversation page for the unread message.
5	Enter the message in the provided text box and press ”Send” button.	The page gets reloaded.
6	The page is completely loaded.	The message appears in the conversation section of the page.

Posting

The following subsection refers to test cases using the posting system of a normal user.

Publish a public post

Table 69: Test case: Publish a public post

Test case name: Publish a public post		
This test case refers to the use case described in table 16 on page 28. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Enter your status in the ”Update your status here” textfield.	
2	Select ”Public Post” from the drop down menu.	
3	Press the ”Post” button.	The page gets reloaded.
4	The page is completely loaded.	The post appears on the top of the list of posts as a public post.

Publish a private post

Table 70: Test case: Publish a private post

Test case name: Publish a private post		
This test case refers to the use case described in table 16 on page 28. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Enter your status in the "Update your status here" textfield.	
2	Select "Private Post" from the drop down menu.	
3	Press the "Post" button.	The page gets reloaded.
4	The page is completely loaded.	The post appears on the top of the list of posts as a private post.

Publish an empty post

Table 71: Test case: Publish an empty post

Test case name: Publish an empty post		
This test case refers to the use case described in table 16 on page 28. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Enter nothing in the "Update your status here" textfield.	
2	Select "Private Post" or "Public Post" from the drop down menu.	
3	Press the "Post" button.	The page gets reloaded.
4	The page is completely loaded.	The system notifies the user that the status was not posted because it was empty.

Edit a post

Table 72: Test case: Edit a post

Test case name: Edit a post		
This test case refers to the use case described in table 21 on page 34. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to edit (from profile or home page).	
2	Press the ”Edit” button.	The page gets reloaded.
3	The page is completely loaded.	The page provides a textfield where the post can be edited.
4	Edit the post.	
5	Press the ”Save” button.	The page gets reloaded.
6	The page is completely loaded.	The page shows the edited post.

Delete a post

Table 73: Test case: Delete a post

Test case name: Delete a post		
This test case refers to the use case described in table 22 on page 35. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to delete (from profile or home page).	
2	Press the ”Delete” button.	The page gets reloaded.
3	The page is completely loaded.	The page does not show the deleted post anymore.

Up vote a post

Table 74: Test case: Up vote a post

Test case name: Up vote a post		
This test case refers to the use case described in table 20 on page 33. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to up vote (from profile or home page).	
2	Press the ”Up-vote” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the post, his ”Karma” counter is increased by one.

Down vote a post

Table 75: Test case: Down vote a post

Test case name: Down vote a post		
This test case refers to the use case described in table 20 on page 33. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to up vote (from profile or home page).	
2	Press the ”Down-vote” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the post, his ”Karma” counter is decreased by one.

Commenting

The following subsection refers to test cases using the commenting system of a normal user.

Comment on post

Table 76: Test case: Comment on post

Test case name: Comment on post		
This test case refers to the use case described in table 17 on page 29. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to comment on (from profile or home page).	
2	Enter your comment in the provided text box.	
3	Press the "Comment" button.	The page gets reloaded.
4	The page is completely loaded.	The page shows the post with your new comment beneath it.

Publish an empty comment

Table 77: Test case: Publish an empty comment

Test case name: Publish an empty comment		
This test case refers to the use case described in table 17 on page 29. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to comment on (from profile or home page).	
2	Enter nothing in the provided text box.	

Continued on next page

Table 77 – continued from previous page

Step Nr.	Step Description	Expected result
3	Press the "Comment" button.	The page gets reloaded.
4	The page is completely loaded.	The system notifies you that the comment was empty and not published.

Edit a comment

Table 78: Test case: Edit a comment

Test case name: Edit a post		
This test case refers to the use case described in table 21 on page 34. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the comment you want to edit.	
2	Press the "Edit" button.	The page gets reloaded.
3	The page is completely loaded.	The page provides a textfield where the comment can be edited.
4	Edit the comment.	
5	Press the "Save" button.	The page gets reloaded.
6	The page is completely loaded.	The page shows the post with the edited comment.

Delete a comment

Table 79: Test case: Delete a comment

Test case name: Delete a comment		
This test case refers to the use case described in table 22 on page 35. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the comment you want to delete.	
2	Press the ”Delete” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the post without the deleted comment.

Up vote a comment

Table 80: Test case: Up vote a comment

Test case name: Up vote a comment		
This test case refers to the use case described in table 20 on page 33. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the comment you want to up vote.	
2	Press the ”Up-vote” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the comment, his ”Karma” counter is increased by one.

Down vote a comment

Table 81: Test case: Down vote a comment

Test case name: Down vote a comment		
This test case refers to the use case described in table 20 on page 33. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the comment you want to down vote.	
2	Press the ”Down-vote” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the comment, his ”Karma” counter is decreased by one.

Notifications

The following test case uses the notification system of a normal user.

Table 82: Test case: View notifications

Test case name: View notifications		
This test case refers to the use case described in table 19 on page 32. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Notifications” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows a list of all recent and new notifications.
3	Select a notification.	The page gets reloaded.
4	The page is completely loaded.	The page shows the post belonging to the notification.

Profile management

The following subsection refers to test cases using the profile management system of a normal user.

Update personal information

Table 83: Test case: Update personal information

Test case name: Update personal information		
This test case refers to the use case described in table 24 on page 38. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Change several information using only valid input.	
4	Press on the ”Save” button.	The page gets reloaded.
5	The page is completely loaded.	The page shows the updated information.

Discard updated personal information

Table 84: Test case: Discard updated personal information

Test case name: Discard updated personal information		
This test case refers to the use case described in table 24 on page 38. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Change several information using only valid input.	

Continued on next page

Table 84 – continued from previous page

Step Nr.	Step Description	Expected result
4	Press on the ”Discard” button.	The page gets reloaded.
5	The page is completely loaded.	The page shows the original –unchanged– information.

Update personal information using invalid input

Table 85: Test case: Update personal information using invalid input

Test case name: Update personal information using invalid input		
This test case refers to the use case described in table 24 on page 38. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Change several information using invalid input some- or everywhere.	
4	Press on the ”Save” button.	The page gets reloaded.
5	The page is completely loaded.	The page shows the updated information only where the input was valid. The system also notifies the user that several changes have been invalid..

Change E-mail address

Table 86: Test case: Change E-mail address

Test case name: Change E-mail address		
This test case refers to the use case described in table 27 on page 41. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Change the E-mail address using a valid address.	
4	Press on the "Save" button.	The page gets reloaded.
5	The page is completely loaded.	The page shows the updated E-mail.
6	Log out of the system (See table 49 on page 75).	The user sees the Log in page.
7	Log in using the new E-mail address.	The system successfully logs the user in.

Change E-mail address using invalid input

Table 87: Test case: Change E-mail address using invalid input

Test case name: Change E-mail address using invalid input		
This test case refers to the use case described in table 27 on page 41. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Change the E-mail address using an invalid address.	

Continued on next page

Table 87 – continued from previous page

Step Nr.	Step Description	Expected result
4	Press on the ”Save” button.	The page gets reloaded.
5	The page is completely loaded.	The E-mail address is not updated and the system notifies the user that the address has been invalid.

Change password

Table 88: Test case: Change password

Test case name: Change password		
This test case refers to the use case described in table 28 on page 43. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Enter the correct password in the ”Old Password” text field.	
4	Enter the new password in the ”New Password” and ”Retype Password” field.	
5	Press on the ”Save” button.	The page gets reloaded.
6	The page is completely loaded.	The page notifies that the password was successfully changed.
7	Log out of the system (See table 49 on page 75).	The user sees the Log in page.
8	Log in using the new password.	The system successfully logs the user in.

Change password using invalid credentials

Table 89: Test case: Change password using invalid credentials

Test case name: Change password using invalid credentials		
This test case refers to the use case described in table 28 on page 43. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Enter the incorrect password in the "Old Password" text field.	
4	Enter the new password in the "New Password" and "Retype Password" field.	
5	Press on the "Save" button.	The page gets reloaded.
6	The page is completely loaded.	The page notifies that the password was not changed, because the old password has been wrong.

Change password using invalid input

Table 90: Test case: Change password using invalid input

Test case name: Change password using invalid input		
This test case refers to the use case described in table 28 on page 43. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.

Continued on next page

Table 90 – continued from previous page

Step Nr.	Step Description	Expected result
3	Enter the correct password in the ”Old Password” text field.	
4	Enter the new password in the ”New Password” field and a different phrase in the ”Retype Password” field or leave one of them empty.	
5	Press on the ”Save” button.	The page gets reloaded.
6	The page is completely loaded.	The page notifies that the password not changed, because the new passwords are not matching or are invalid.

Change profile picture

Table 91: Test case: Change profile picture

Test case name: Change profile picture		
This test case refers to the use case described in table 26 on page 40. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Press the ”Change picture” button.	The system asks for the path of the new picture.
4	Select a picture from your computer.	The system starts the upload and refreshes the page.
5	The page is completely loaded.	The new picture is shown.

Change profile picture using an unsupported format

Table 92: Test case: Change profile picture using an unsupported format

Test case name: Change profile picture using an unsupported format		
This test case refers to the use case described in table 26 on page 40. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Press the ”Change picture” button.	The system asks for the path of the new picture.
4	Select a picture with an unsupported format from your computer.	The system refreshes the page.
5	The page is completely loaded.	The system notifies the user that the picture format is not supported.

Change profile picture using a picture exceeding the maximum size

Table 93: Test case: Change profile picture using a picture exceeding the maximum size

Test case name: Change profile picture using a picture exceeding the maximum size		
This test case refers to the use case described in table 26 on page 40. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.

Continued on next page

Table 93 – continued from previous page

Step Nr.	Step Description	Expected result
3	Press the "Change picture" button.	The system asks for the path of the new picture.
4	Select a picture which exceeds the maximum size from your computer.	The system refreshes the page.
5	The page is completely loaded.	The system notifies the user that the picture's size is too big.

Remove connected administrator/fanpage profile

Table 94: Test case: Remove connected administrator/fanpage profile

Test case name: Remove connected administrator/fanpage profile		
This test case refers to the use case described in table 29 on page 44. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Press the "Remove" button next to the connected profile you want to remove.	The page gets reloaded.
4	The page is completely loaded.	The system shows the profile editing page without the recent deleted account.

Delete account

Table 95: Test case: Delete account

Test case name: Delete account		
This test case refers to the use case described in table 30 on page 45. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the profile editing page.
3	Press the ”Delete account” button.	The system asks for confirmation to delete the account.
4	You confirm the deletion.	The page gets reloaded.
5	The page is completely loaded.	The system shows the log in page.
6	Enter your credentials to log in.	The page gets reloaded.
7	The page is completely loaded.	The system notifies you that the provided credentials are wrong.

3.10.3 Fanpage admin

The following sections refers to all test cases using a fanpage administrator profile. All test cases require that a fanpage administrator is logged into the system.

View homepage

Table 96: Test case: View homepage

Test case name: View homepage		
This test case refers to the use case described in table 23 on page 37. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the ”The Network” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents all recent posts of your fanpage in the middle of the page. The header item has a badge if there are new notifications. The header shows the amount of people following your page.

Posting

The following subsection refers to test cases using the posting system of a fanpage administrator.

Publish a public post

Table 97: Test case: Publish a public post

Test case name: Publish a public post		
This test case refers to the use case described in table 16 on page 28. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Enter your status in the ”Update your status here” textfield.	
2	Press the ”Post” button.	The page gets reloaded.
3	The page is completely loaded.	The post appears on the top of the list of posts as a public post.

Publish an empty post

Table 98: Test case: Publish an empty post

Test case name: Publish an empty post		
This test case refers to the use case described in table 16 on page 28. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Enter nothing in the ”Update your status here” textfield.	
2	Press the ”Post” button.	The page gets reloaded.
3	The page is completely loaded.	The system notifies the user that the status was not posted because it was empty.

Edit a post

Table 99: Test case: Edit a post

Test case name: Edit a post		
This test case refers to the use case described in table 21 on page 34. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to edit.	
2	Press the ”Edit” button.	The page gets reloaded.
3	The page is completely loaded.	The page provides a textfield where the post can be edited.
4	Edit the post.	
5	Press the ”Save” button.	The page gets reloaded.
6	The page is completely loaded.	The page shows the edited post.

Delete a post

Table 100: Test case: Delete a post

Test case name: Delete a post		
This test case refers to the use case described in table 22 on page 35. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to delete.	
2	Press the ”Delete” button.	The page gets reloaded.
3	The page is completely loaded.	The page does not show the deleted post anymore.

Up vote a post

Table 101: Test case: Up vote a post

Test case name: Up vote a post		
This test case refers to the use case described in table 20 on page 33. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to up vote.	
2	Press the ”Up-vote” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the post, his ”Karma” counter is increased by one.

Down vote a post

Table 102: Test case: Down vote a post

Test case name: Down vote a post		
This test case refers to the use case described in table 20 on page 33. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to down vote.	
2	Press the ”Down-vote” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the post, his ”Karma” counter is decreased by one.

Commenting

The following subsection refers to test cases using the commenting system of a fanpage administrator.

Comment on post

Table 103: Test case: Comment on post

Test case name: Comment on post		
This test case refers to the use case described in table 17 on page 29. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to comment on.	
2	Enter your comment in the provided text box.	
3	Press the ”Comment” button.	The page gets reloaded.
4	The page is completely loaded.	The page shows the post with your new comment beneath it.

Publish an empty comment

Table 104: Test case: Publish an empty comment

Test case name: Publish an empty comment		
This test case refers to the use case described in table 17 on page 29. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the post you want to comment on.	
2	Enter nothing in the provided text box.	
3	Press the ”Comment” button.	The page gets reloaded.

Continued on next page

Table 104 – continued from previous page

Step Nr.	Step Description	Expected result
4	The page is completely loaded.	The system notifies you that the comment was empty and not published.

Edit a comment

Table 105: Test case: Edit a comment

Test case name: Edit a post		
This test case refers to the use case described in table 21 on page 34. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the comment you want to edit.	
2	Press the ”Edit” button.	The page gets reloaded.
3	The page is completely loaded.	The page provides a textfield where the comment can be edited.
4	Edit the comment.	
5	Press the ”Save” button.	The page gets reloaded.
6	The page is completely loaded.	The page shows the post with the edited comment.

Delete a comment

Table 106: Test case: Delete a comment

Test case name: Delete a comment		
This test case refers to the use case described in table 22 on page 35. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the comment you want to delete.	
2	Press the ”Delete” button.	The page gets reloaded.

Continued on next page

Table 106 – continued from previous page

Step Nr.	Step Description	Expected result
3	The page is completely loaded.	The page shows the post without the deleted comment.

Up vote a comment

Table 107: Test case: Up vote a comment

Test case name: Up vote a comment		
This test case refers to the use case described in table 20 on page 33. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the comment you want to up vote.	
2	Press the ”Up-vote” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the comment, his ”Karma” counter is increased by one.

Down vote a comment

Table 108: Test case: Down vote a comment

Test case name: Down vote a comment		
This test case refers to the use case described in table 20 on page 33. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the comment you want to up vote.	
2	Press the ”Down-vote” button.	The page gets reloaded.
3	The page is completely loaded.	The page shows the comment, his ”Karma” counter is decreased by one.

Notifications

The following test case uses the notification system of a fanpage administrator.

Table 109: Test case: View notifications

Test case name: View notifications		
This test case refers to the use case described in table 19 on page 32. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Notifications" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows a list of all recent and new notifications.
3	Select a notification.	The page gets reloaded.
4	The page is completely loaded.	The page shows the post belonging to the notification.

Profile management

The following subsection refers to test cases using the profile management system of a normal user.

Update page information

Table 110: Test case: Update page information

Test case name: Update page information		
This test case refers to the use case described in table 25 on page 39. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.

Continued on next page

Table 110 – continued from previous page

Step Nr.	Step Description	Expected result
3	Change several information using only valid input.	
4	Press on the "Save" button.	The page gets reloaded.
5	The page is completely loaded.	The page shows the updated information.

Discard updated page information

Table 111: Test case: Discard updated page information

Test case name: Discard updated page information		
This test case refers to the use case described in table 25 on page 39. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Change several information using only valid input.	
4	Press on the "Discard" button.	The page gets reloaded.
5	The page is completely loaded.	The page shows the original –unchanged– information.

Update page information using invalid input

Table 112: Test case: Update page information using invalid input

Test case name: Update page information using invalid input		
This test case refers to the use case described in table 25 on page 39. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Change several information using invalid input some- or everywhere.	
4	Press on the "Save" button.	The page gets reloaded.
5	The page is completely loaded.	The page shows the updated information only where the input was valid. The system also notifies the user that several changes have been invalid..

Change E-mail address

Table 113: Test case: Change E-mail address

Test case name: Change E-mail address		
This test case refers to the use case described in table 27 on page 41. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.

Continued on next page

Table 113 – continued from previous page

Step Nr.	Step Description	Expected result
3	Change the E-mail address using a valid address.	
4	Press on the "Save" button.	The page gets reloaded.
5	The page is completely loaded.	The page shows the updated E-mail.
6	Log out of the system (See table 49 on page 75).	The user sees the Log in page.
7	Log in using the new E-mail address.	The system successfully logs the user in.

Change E-mail address using invalid input

Table 114: Test case: Change E-mail address using invalid input

Test case name: Change E-mail address using invalid input		
This test case refers to the use case described in table 27 on page 41. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the "Edit" button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Change the E-mail address using an invalid address.	
4	Press on the "Save" button.	The page gets reloaded.
5	The page is completely loaded.	The E-mail address is not updated and the system notifies the user that the address has been invalid.

Change password

Table 115: Test case: Change password

Test case name: Change password		
This test case refers to the use case described in table 28 on page 43. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Enter the correct password in the ”Old Password” text field.	
4	Enter the new password in the ”New Password” and ”Retype Password” field.	
5	Press on the ”Save” button.	The page gets reloaded.
6	The page is completely loaded.	The page notifies that the password was successfully changed.
7	Log out of the system (See table 49 on page 75).	The user sees the Log in page.
8	Log in using the new password.	The system successfully logs the user in.

Change password using invalid credentials

Table 116: Test case: Change password using invalid credentials

Test case name: Change password using invalid credentials		
This test case refers to the use case described in table 28 on page 43. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Enter the incorrect password in the ”Old Password” text field.	
4	Enter the new password in the ”New Password” and ”Retype Password” field.	
5	Press on the ”Save” button.	The page gets reloaded.
6	The page is completely loaded.	The page notifies the user that the password was not changed, because the old password has been wrong.

Change password using invalid input

Table 117: Test case: Change password using invalid input

Test case name: Change password using invalid input		
This test case refers to the use case described in table 28 on page 43. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.

Continued on next page

Table 117 – continued from previous page

Step Nr.	Step Description	Expected result
3	Enter the correct password in the ”Old Password” text field.	
4	Enter the new password in the ”New Password” field and a different phrase in the ”Retype Password” field or leave one of them empty.	
5	Press on the ”Save” button.	The page gets reloaded.
6	The page is completely loaded.	The page notifies the user that the password was not changed, because the new passwords are not matching or are invalid.

Change fanpage picture

Table 118: Test case: Change fanpage picture

Test case name: Change fanpage picture		
This test case refers to the use case described in table 26 on page 40. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Press the ”Change picture” button.	The system asks for the path of the new picture.
4	Select a picture from your computer.	The system starts the upload and refreshes the page.
5	The page is completely loaded.	The new picture is shown.

Change fanpage picture using an unsupported format

Table 119: Test case: Change fanpage picture using an unsupported format

Test case name: Change fanpage picture using an unsupported format		
This test case refers to the use case described in table 26 on page 40. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Press the ”Change picture” button.	The system asks for the path of the new picture.
4	Select a picture with an unsupported format from your computer.	The system refreshes the page.
5	The page is completely loaded.	The system notifies the user that the picture format is not supported.

Change fanpage picture using a picture exceeding the maximum size

Table 120: Test case: Change fanpage picture using a picture exceeding the maximum size

Test case name: Change fanpage picture using a picture exceeding the maximum size		
This test case refers to the use case described in table 26 on page 40. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.

Continued on next page

Table 120 – continued from previous page

Step Nr.	Step Description	Expected result
3	Press the ”Change picture” button.	The system asks for the path of the new picture.
4	Select a picture which exceeds the maximum size from your computer.	The system refreshes the page.
5	The page is completely loaded.	The system notifies the user that the picture’s size is too big.

Connect fanpage to user

Table 121: Test case: Connect fanpage to user

Test case name: Connect fanpage to user		
This test case refers to the use case described in table 32 on page 48. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Enter the valid E-mail address of the user in the text box next to the ”Connect user account” label.	
4	Press the ”Save” button.	The page gets reloaded.
5	The page is completely loaded.	The page shows a notification that the user has been added as the fanpage administrator.

Connect fanpage to user using an invalid e-mail

Table 122: Test case: Connect fanpage to user using an invalid e-mail

Test case name: Connect fanpage to user using an invalid e-mail		
This test case refers to the use case described in table 32 on page 48. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.
3	Enter an invalid E-mail addressin the text box next to the ”Connect user account” label.	
4	Press the ”Save” button.	The page gets reloaded.
5	The page is completely loaded.	The page shows a notification that the E-mail address has been invalid and the user has not been added as administrator.

Delete fanpage

Table 123: Test case: Delete fanpage

Test case name: Delete fanpage		
This test case refers to the use case described in table 30 on page 45. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Select the ”Edit” button on the top of the page.	The page gets reloaded.
2	The page is completely loaded.	The page shows the fanpage editing page.

Continued on next page

Table 123 – continued from previous page

Step Nr.	Step Description	Expected result
3	Press the ”Delete fanpage” button.	The system asks for confirmation to delete the fanpage.
4	You confirm the deletion.	The page gets reloaded.
5	The page is completely loaded.	The system shows the log in page.
6	Enter your credentials to log in.	The page gets reloaded.
7	The page is completely loaded.	The system notifies you that the provided credentials are wrong.

3.10.4 System administrator

The following sections refers to all test cases using a system administrator profile. All test cases require that a system administrator is logged into the system.

View homepage

Table 124: Test case: View homepage

Test case name: View homepage		
Pre-condition: The system administrator is logged in.		
Step Nr.	Step Description	Expected result
1	Press on the ”The Network” button on the top of the page	The page gets refreshed.
2	The page is completely loaded.	The system presents several statistics about the service in the middle of the page.

Manage user

The following section refers to all test cases using the user management system of a system administrator.

Find user

Table 125: Test case: Find user

Test case name: Find user		
Pre-condition: The system administrator is logged in.		
Step Nr.	Step Description	Expected result
1	Press on the "Manage user" button on the top of the page.	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all user using the service.
3	Enter the name of a user you want to find in the search box on the top of the page and click on the search button.	The pages gets refreshed.
4	The page is completely loaded.	The system presents a list of all user fitting to the search query.

Reset user password

Table 126: Test case: Reset user password

Test case name: Reset user password		
This test case refers to the use case described in table 31 on page 47. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the "Manage user" button on the top of the page.	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all users using the service.

Continued on next page

Table 126 – continued from previous page

Step Nr.	Step Description	Expected result
3	Select the user you want to reset the password. Press on the "Reset Password" button next to his name.	The system reloads the page and the user gets an E-mail with his new password.

Connect to user

Table 127: Test case: Connect to user

Test case name: Connect to user		
This test case refers to the use case described in table 32 on page 48. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the "Manage user" button on the top of the page.	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all user using the service.
3	Select the user you want to connect to the profile. Press on the "Connect with user" button next to his name.	The system reloads the page and the user gets connected to the profile.

Delete user

Table 128: Test case: Delete user

Test case name: Delete user		
This test case refers to the use case described in table 30 on page 45. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the "Manage user" button on the top of the page.	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all user using the service.

Continued on next page

Table 128 – continued from previous page

Step Nr.	Step Description	Expected result
3	Select the user you want to delete. Press on the ”Delete user” button next to his name.	The system reloads the page and the user gets deleted. His name disappears from the list.

Manage fanpages

The following section refers to all test cases using the fanpage management system of a system administrator.

Find fanpage

Table 129: Test case: Find fanpage

Test case name: Find fanpage		
Pre-condition: The system administrator is logged in.		
Step Nr.	Step Description	Expected result
1	Press on the ”Manage fanpages” button on the top of the page.	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all fanpages signed in for the service.
3	Enter the name of a fanpage you want to find in the search box on the top of the page and click on the search button.	The pages gets refreshed.
4	The page is completely loaded.	The system presents a list of all fanpages fitting to the search query.

Delete fanpage

Table 130: Test case: Delete fanpage

Test case name: Delete fanpage		
This test case refers to the use case described in table 30 on page 45. All pre- and post-conditions apply.		
Step Nr.	Step Description	Expected result
1	Press on the "Manage fanpages" button on the top of the page.	The page gets refreshed.
2	The page is completely loaded.	The system presents a list of all fanpages signed in for the service.
3	Select the fanpage you want to delete. Press on the "Delete fanpage" button next to his name.	The system reloads the page and the fanpage gets deleted. Its name disappears from the list.

3.11 Java application client

An obligatory requirement is the development of a Java desktop application as a client for the web service. This client will receive objects using a XML-based communication. The client is going to reconstruct an array list of object from this XML file.

With the client a normal user will be able to log into the system and receive the latest posts of friends and fanpages, just as outlined in the use case described in table 23 on page 37.

List of Figures

List of Tables

Listings

Bibliography

[BBC13] BBC. *Twitter shares jump 73% in market debut.* Nov. 2013. URL: http://www.bbc.co.uk/news/business-24851054.

[Coc01] Alistair Cockburn. *Writing Effective Use Cases.* Addison-Wesley, 2001.

[CS14] John Cannarella and Joshua A. Spechler. 'Epidemiological modeling of online social network dynamics'. MA thesis. Department of Mechanical and Aerospace Engineering, Princeton University, Princeton, NJ, USA, Jan. 2014. URL: http://arxiv.org/pdf/1401.4208v1.pdf.

[DB13] Maeve Duggan and Joanna Brenner. 'The Demographics of Social Media Users - 2012'. In: *Pew Research* (Feb. 2013). URL: http://www.pewinternet.org/files/old-media/Files/Reports/2013/PIP_SocialMediaUsers.pdf.

[Dub13] Ryan Dube. *Is Facebook going the same way as Myspace?* Oct. 2013. URL: http://www.makeuseof.com/tag/is-facebook-going-the-same-way-as-myspace/.

[Fow03] Martin Fowler. *Patterns of enterprise application architecture.* Addison-Wesley, 2003.

[Gam+94] Erich Gamma et al. *Design Patterns. Elements of Reusable Object-Oriented Software.* Addison-Wesley, 1994.

[Ger12] Financial Times Germany. 'Facebook Aktie kostet 38 Dollar'. In: *Financial Times Germany* (May 2012). URL: https://web.archive.org/web/20120519015802/http://www.ftd.de/it-medien/medien-internet/:boersengang-facebook-aktien-kosten-38-dollar/70038544.html.

[How+11] Philip N. Howard et al. 'Opening Closed Regimes: What Was the Role of Social Media During the Arab Spring?' In: *Information Technology and Political Islam* (2011). URL: `http://pitpi.org/index.php/2011/09/11/opening-closed-regimes-what-was-the-role-of-social-media-during-the-arab-spring/`.

[Inc13] Facebook Inc. *Facebook Anual Report.* Feb. 2013. URL: `http://investor.fb.com/annual-proxy.cfm`.

[KE06] Alfons Kemper and André Eickler. *Datenbanksysteme.* 6th ed. Oldenbourg, 2006.

[Nea13] Ryan W. Neal. 'WhatsApp, SnapChat And LINE: Why Mobile Messaging Apps Are Taking Teens Away From Facebook'. In: *International Business Times* (2013). URL: `http://www.ibtimes.com/whatsapp-snapchat-line-why-mobile-messaging-apps-are-taking-teens-away-facebook-1464804`.

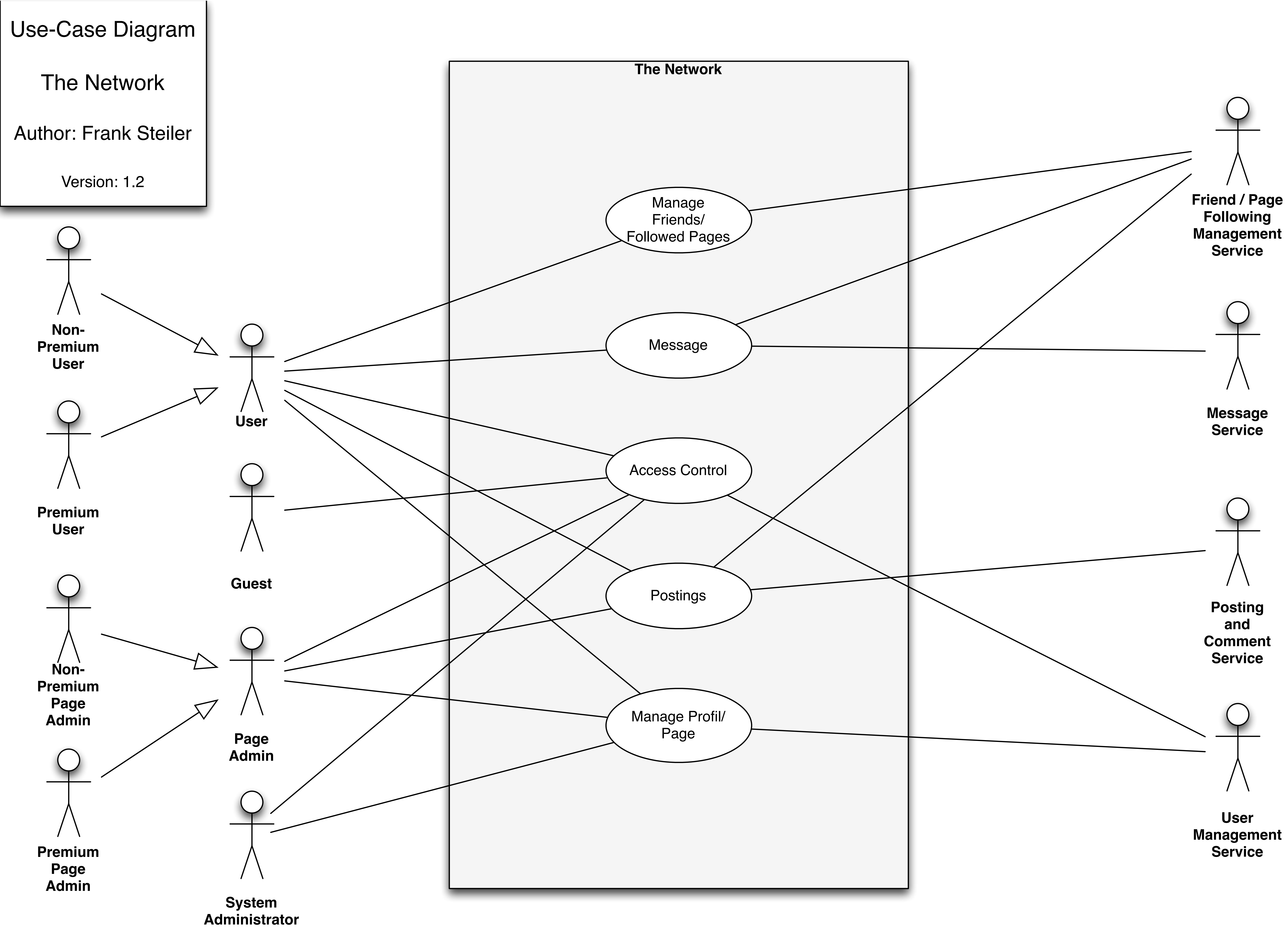
[Nea14] Ryan W. Neal. 'Facebook Gets Older: Demographic Report Shows 3 Million Teens Left Social Network In 3 Years'. In: *International Business Times* (2014). URL: `http://www.ibtimes.com/facebook-gets-older-demographic-report-shows-3-million-teens-left-social-network-3-years-1543092`.

[Sau14] DJ Saul. '3 million teens leave Facebook in 3 years: The 2014 Facebook Demographic Report'. In: *iStrategyLabs* (2014). URL: `http://istrategylabs.com/2014/01/3-million-teens-leave-facebook-in-3-years-the-2014-facebook-demographic-report/`.

[Slo13] Garett Sloane. 'Worries Grow That Facebook Is Overdoing It With Ads How many sponsored posts are too many?' In: *Adweek* (Nov. 2013). URL: `http://www.adweek.com/news/technology/worries-grow-facebook-overdoing-it-ads-154038`.

[Wol12] Micheal Wolff. 'The Facebook Fallacy'. In: *MIT Technology Review* (May 2012). URL: `http://www.technologyreview.com/news/427972/the-facebook-fallacy/`.

Appendices



Use-Case Diagram

The Network

Author: Frank Steiler

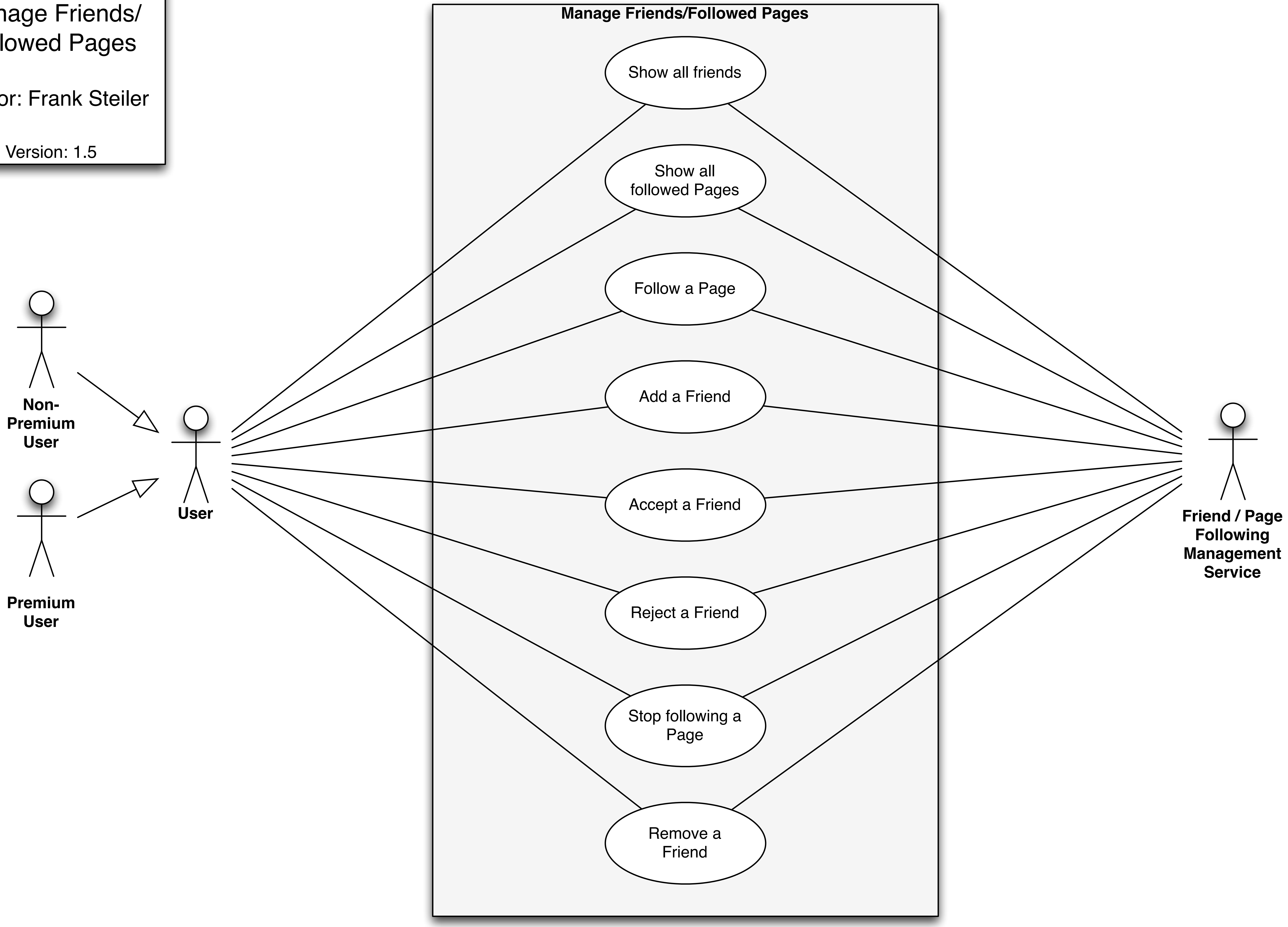
Version: 1.2

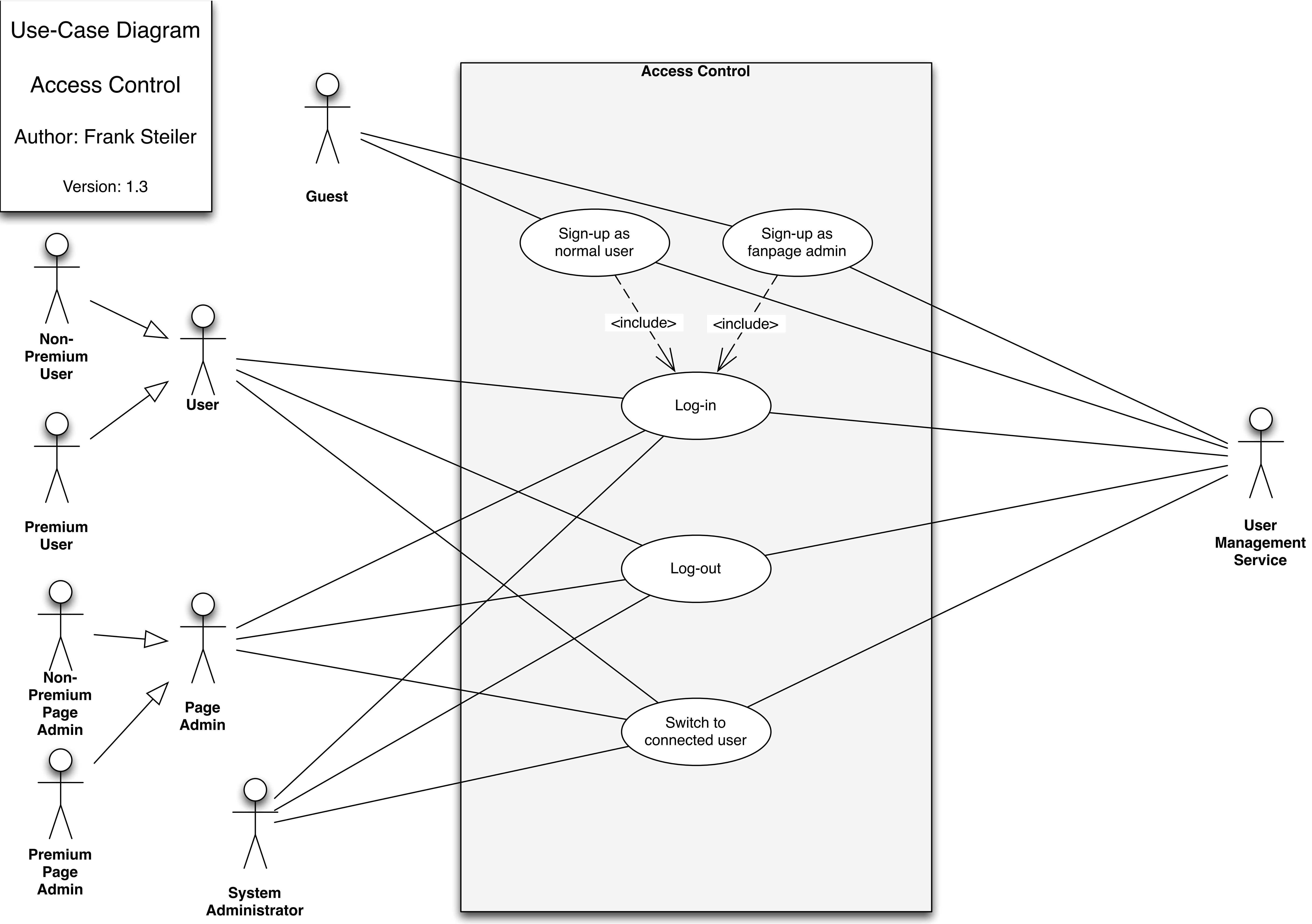
Use-Case Diagram

Manage Friends/
Followed Pages

Author: Frank Steiler

Version: 1.5





Use-Case Diagram

Access Control

Author: Frank Steiler

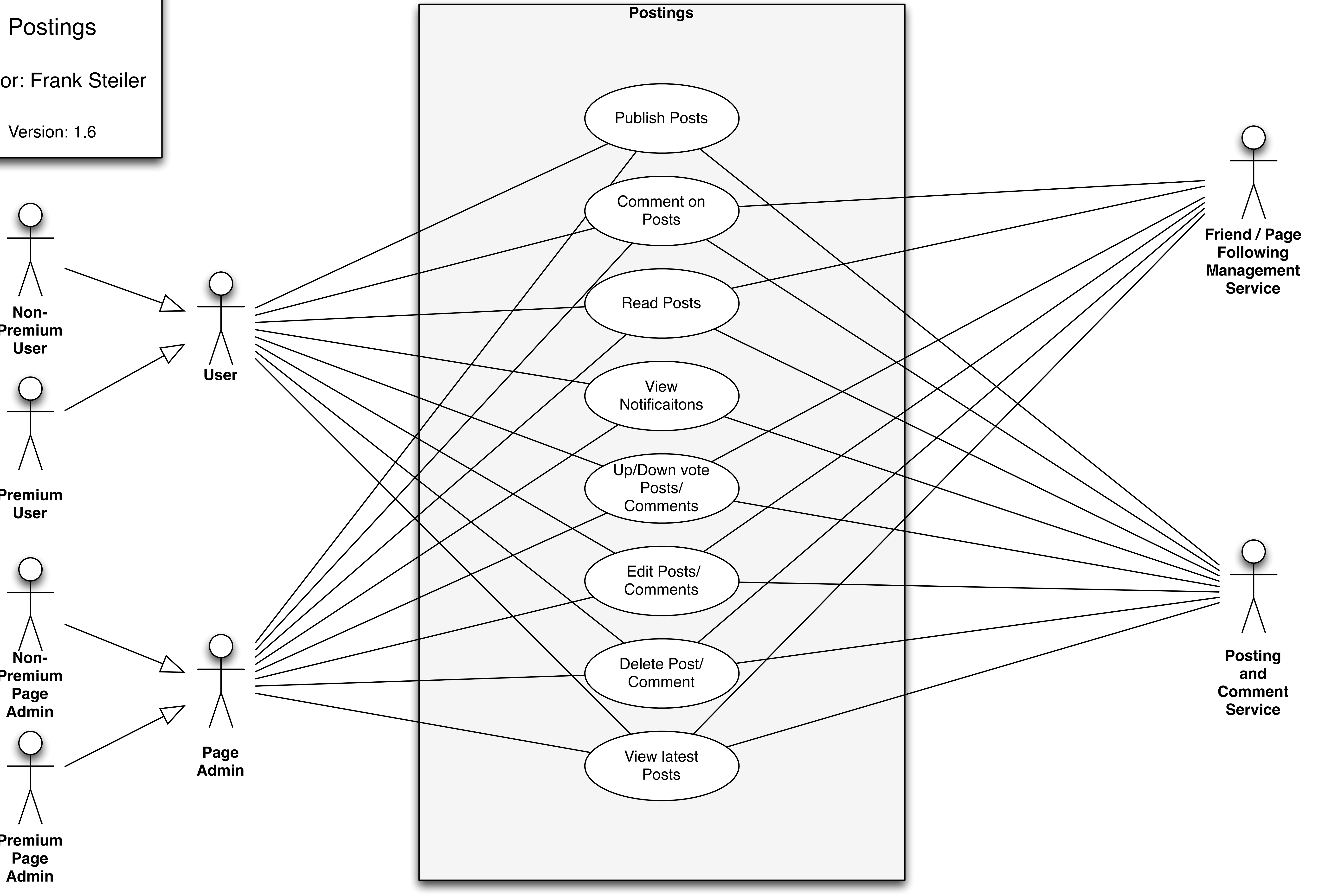
Version: 1.3

Use-Case Diagram

Postings

Author: Frank Steiler

Version: 1.6

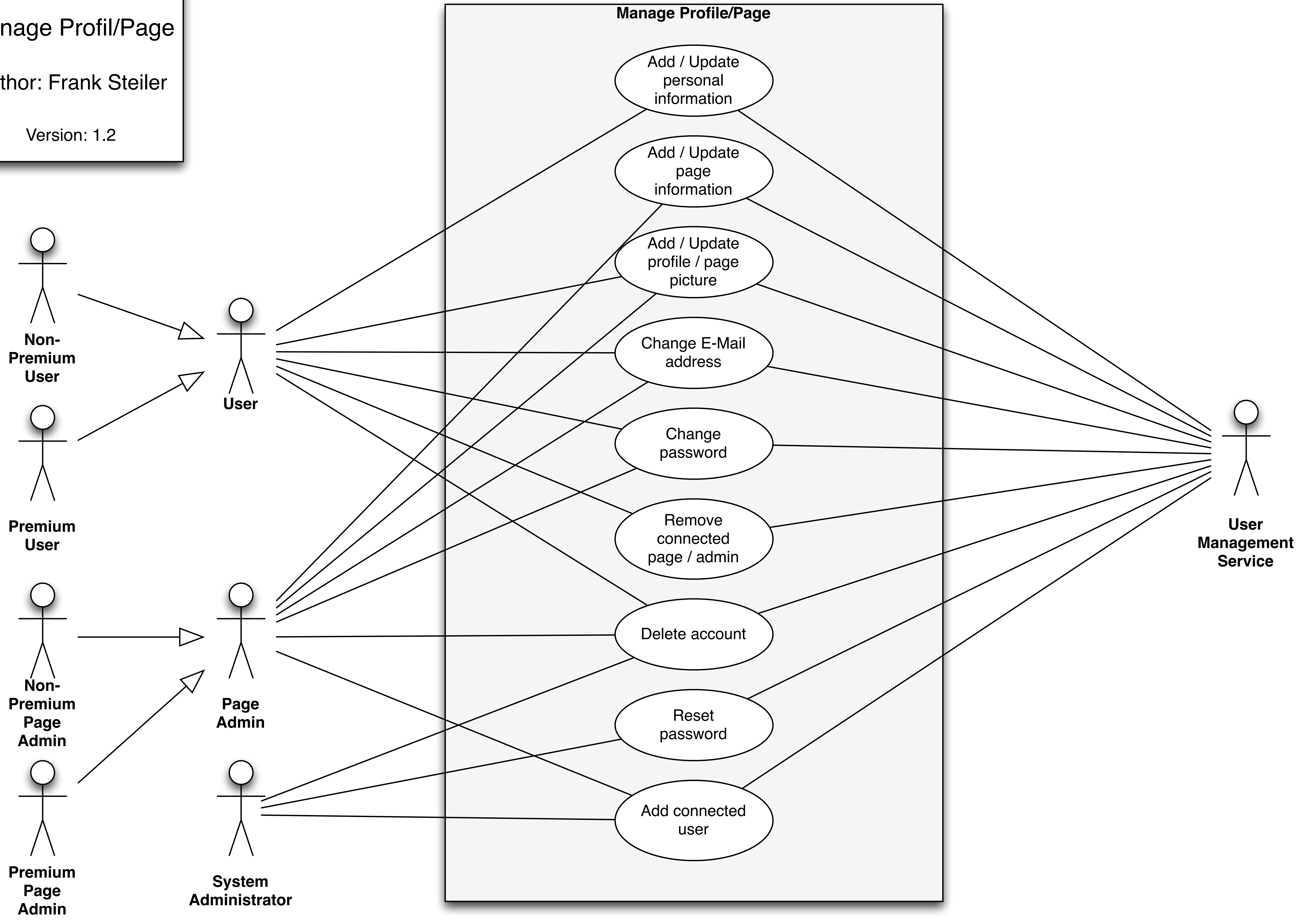


Use-Case Diagram

Manage Profil/Page

Author: Frank Steiler

Version: 1.2

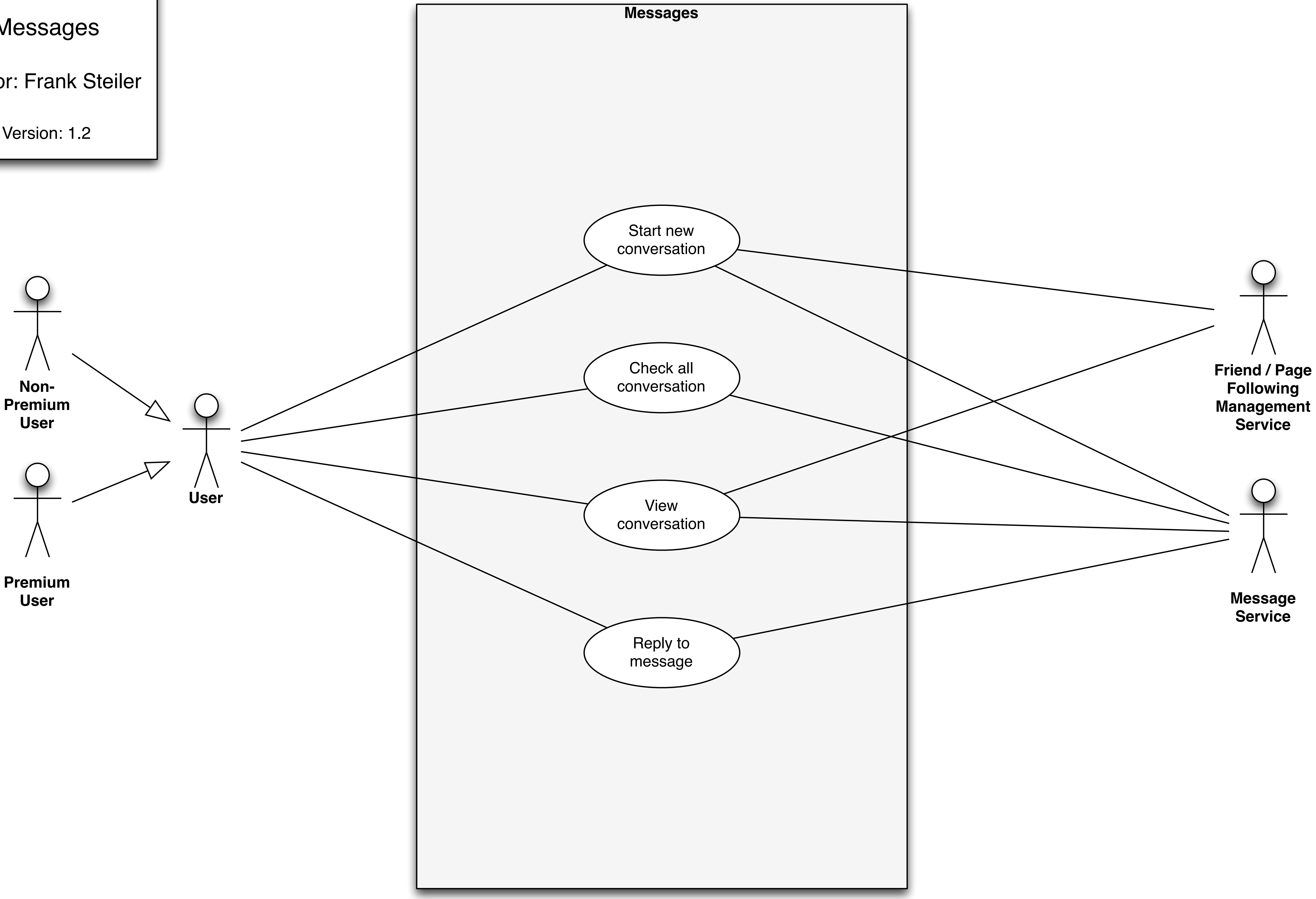


Use-Case Diagram

Messages

Author: Frank Steiler

Version: 1.2

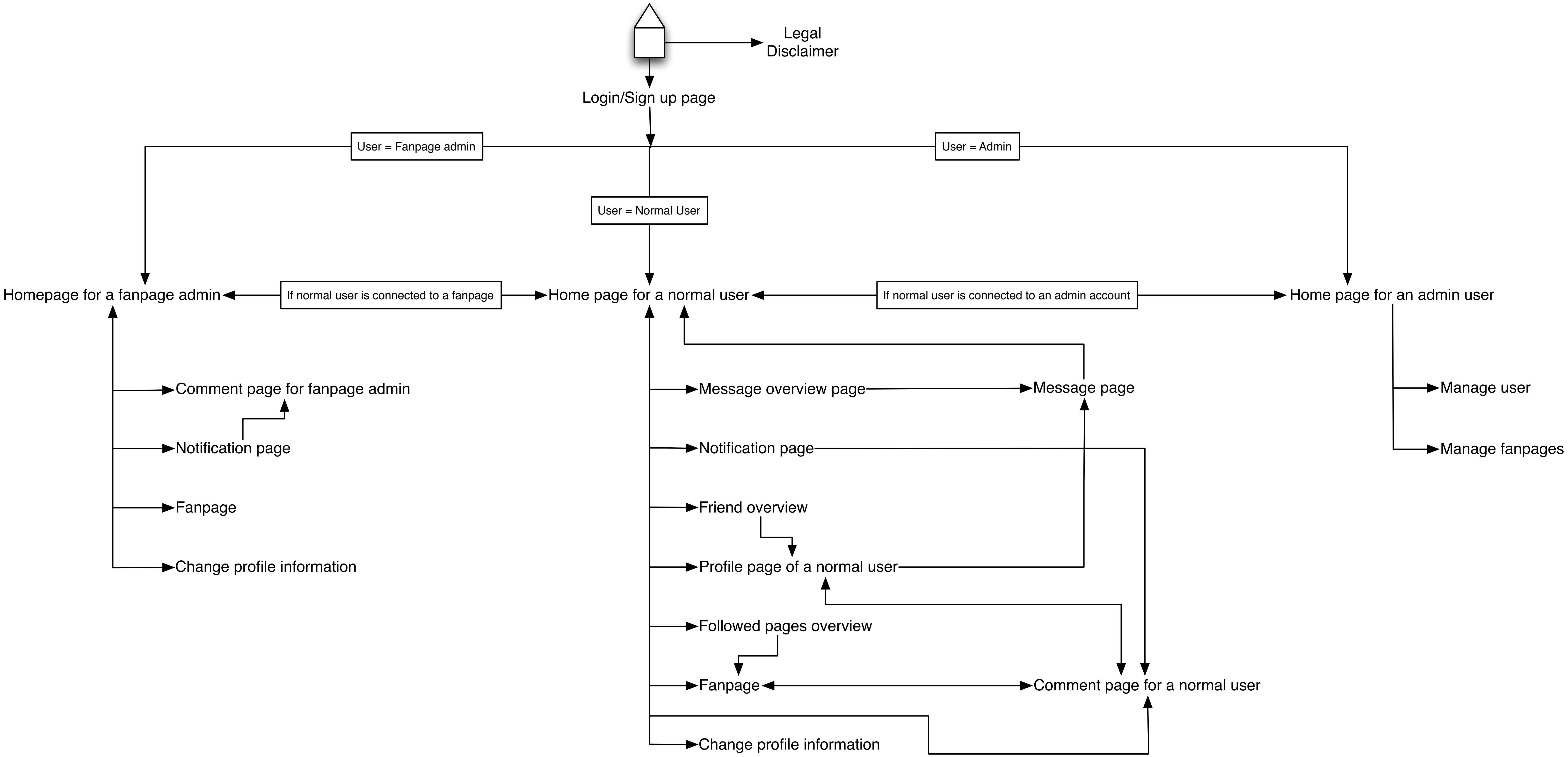


Pagemap

The Network

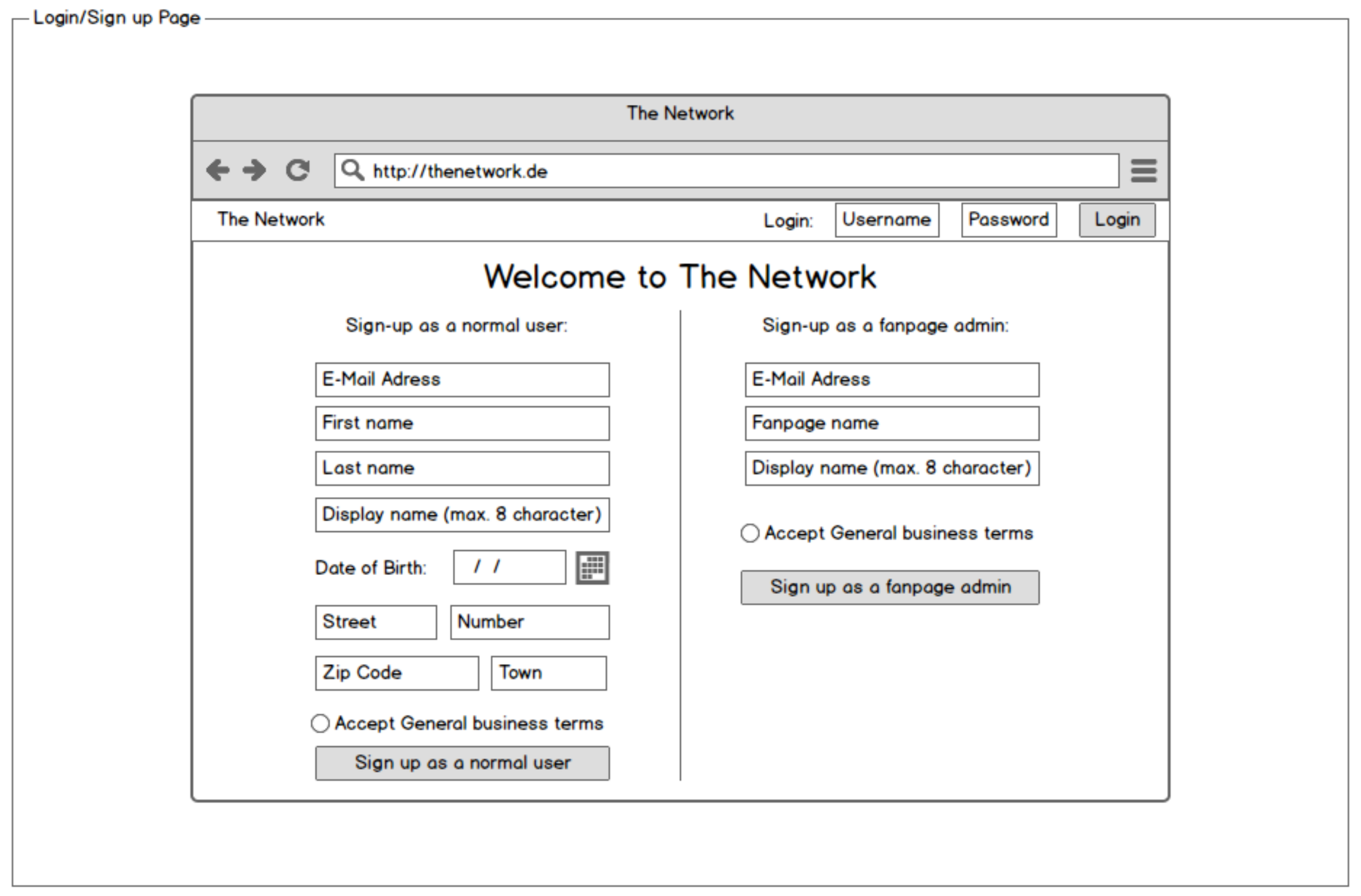
Author: Frank Steiler

Version: 1.1


Legal Disclaimer
Login/Sign up page
User = Fanpage admin
User = Admin
User = Normal User
Homepage for a fanpage admin
If normal user is connected to a fanpage
Home page for a normal user
If normal user is connected to an admin account
Home page for an admin user
Comment page for fanpage admin
Notification page
Fanpage
Change profile information
Message overview page
Message page
Notification page
Friend overview
Profile page of a normal user
Followed pages overview
Fanpage
Comment page for a normal user
Change profile information
Manage user
Manage fanpages

Wireframes for "The Network" by Frank Steiler V 1.6

Login/Sign up Page



The Network

http://thenetwork.de

The Network Login: Username Password Login

Welcome to The Network

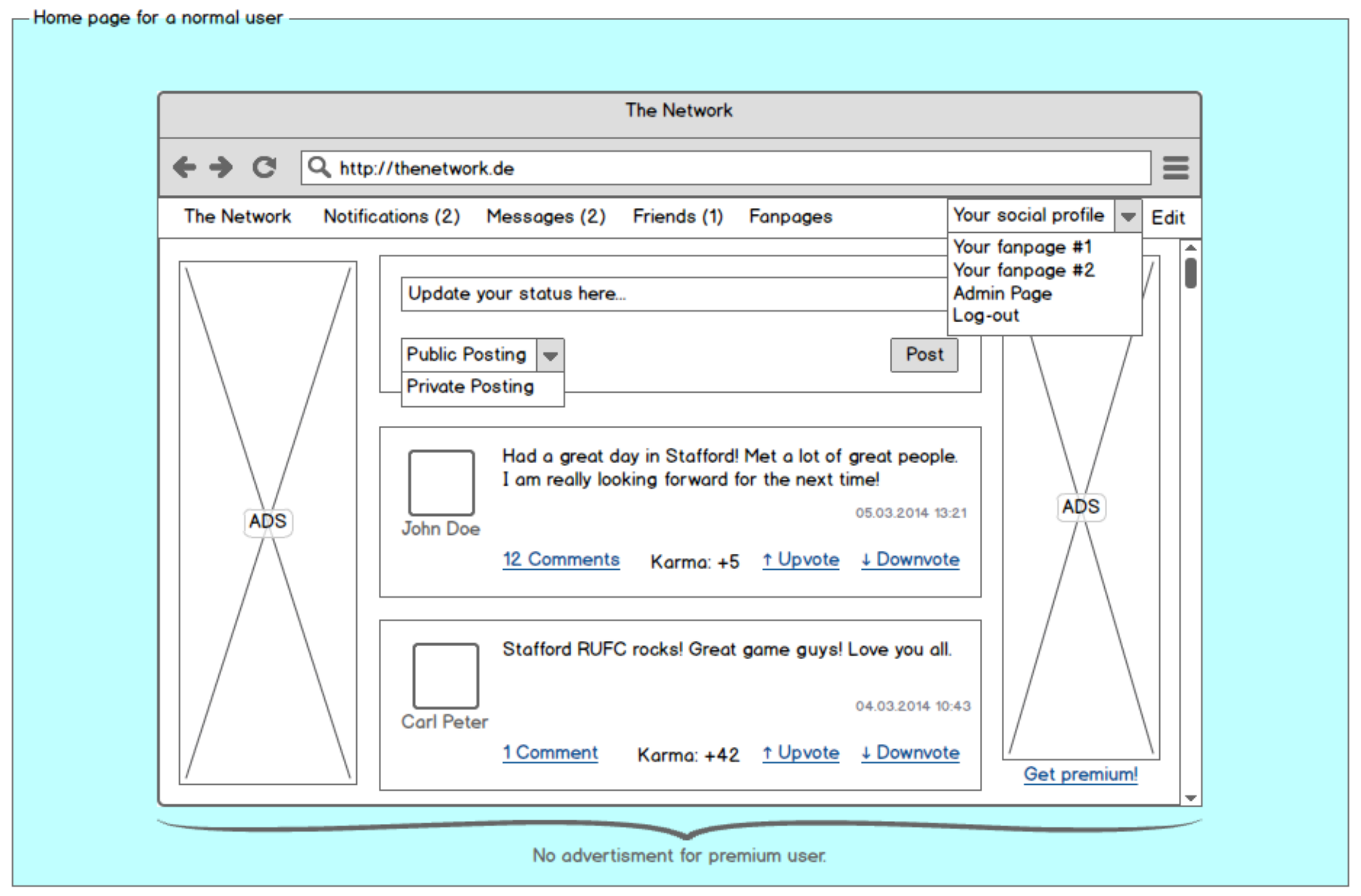
Sign-up as a normal user:

E-Mail Adress
First name
Last name
Display name (max. 8 character)
Date of Birth: / /
Street Number
Zip Code Town
○ Accept General business terms
Sign up as a normal user

Sign-up as a fanpage admin:

E-Mail Adress
Fanpage name
Display name (max. 8 character)
○ Accept General business terms
Sign up as a fanpage admin

Home page for a normal user



The Network

http://thenetwork.de

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

Update your status here...
Public Posting
Private Posting
Post

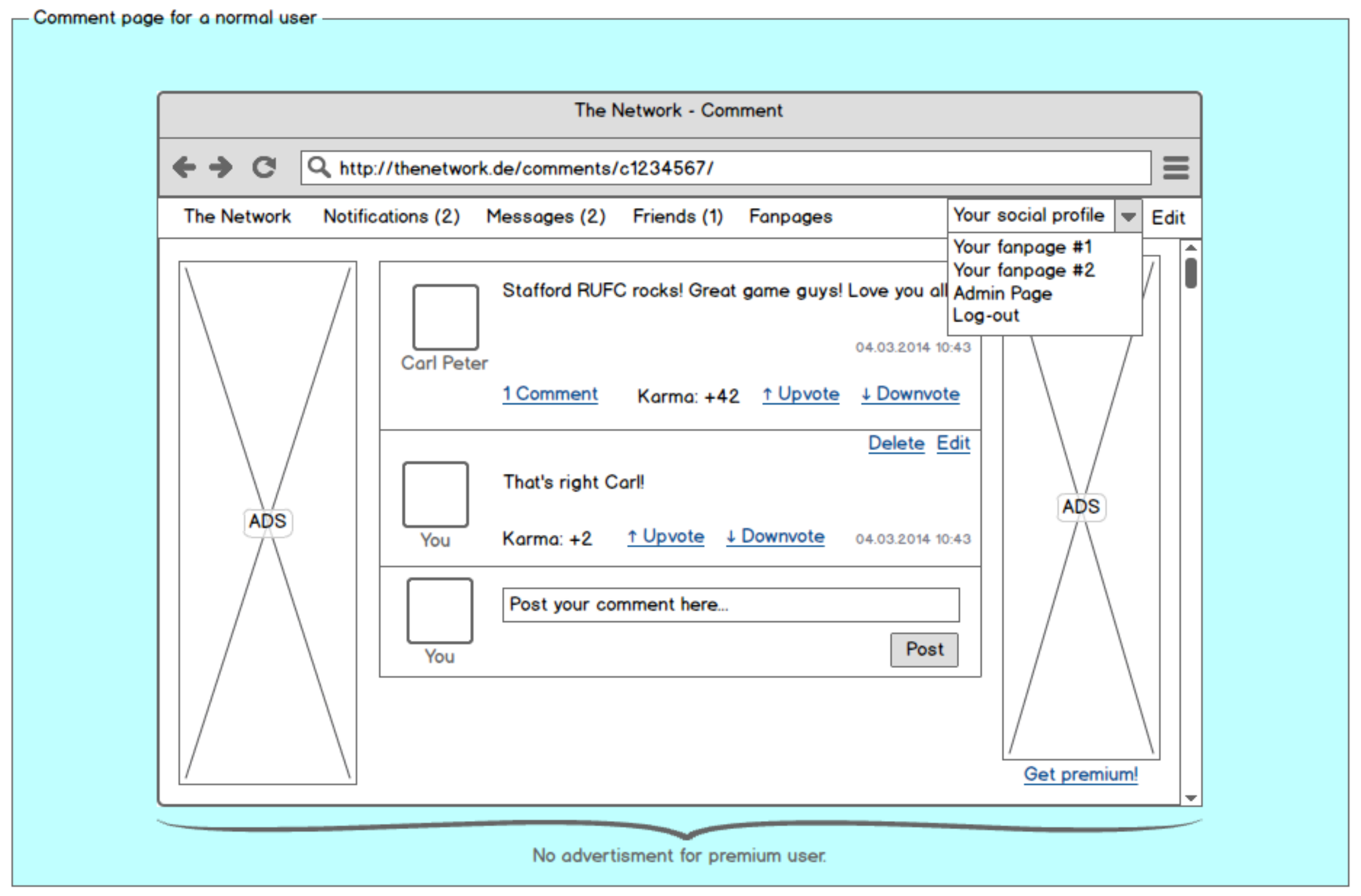
Had a great day in Stafford! Met a lot of great people. I am really looking forward for the next time!
John Doe 05.03.2014 13:21
12 Comments Karma: +5 ↑ Upvote ↓ Downvote

Stafford RUFC rocks! Great game guys! Love you all.
Carl Peter 04.03.2014 10:43
1 Comment Karma: +42 ↑ Upvote ↓ Downvote

ADS ADS Get premium!

No advertisment for premium user.

Comment page for a normal user



The Network - Comment

http://thenetwork.de/comments/c1234567/

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

Stafford RUFC rocks! Great game guys! Love you all
Carl Peter 04.03.2014 10:43
1 Comment Karma: +42 ↑ Upvote ↓ Downvote

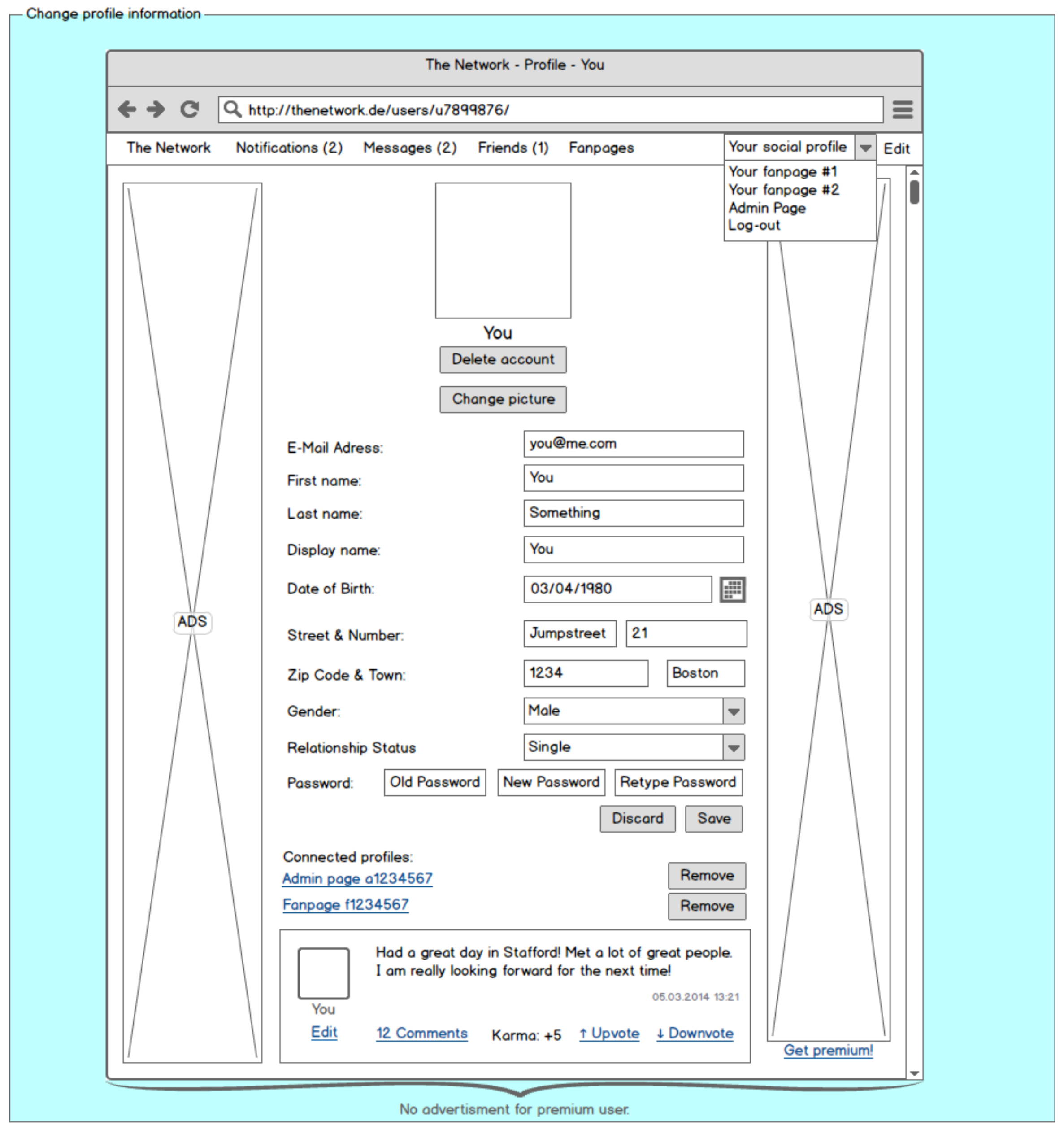
Delete Edit
That's right Carl!
You Karma: +2 ↑ Upvote ↓ Downvote 04.03.2014 10:43

Post your comment here...
You Post

ADS ADS Get premium!

No advertisment for premium user.

Change profile information



The Network - Profile - You

http://thenetwork.de/users/u7899876/

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

You
Delete account
Change picture

E-Mail Adress: you@me.com
First name: You
Last name: Something
Display name: You
Date of Birth: 03/04/1980
Street & Number: Jumpstreet 21
Zip Code & Town: 1234 Boston
Gender: Male
Relationship Status: Single
Password: Old Password New Password Retype Password
Discard Save

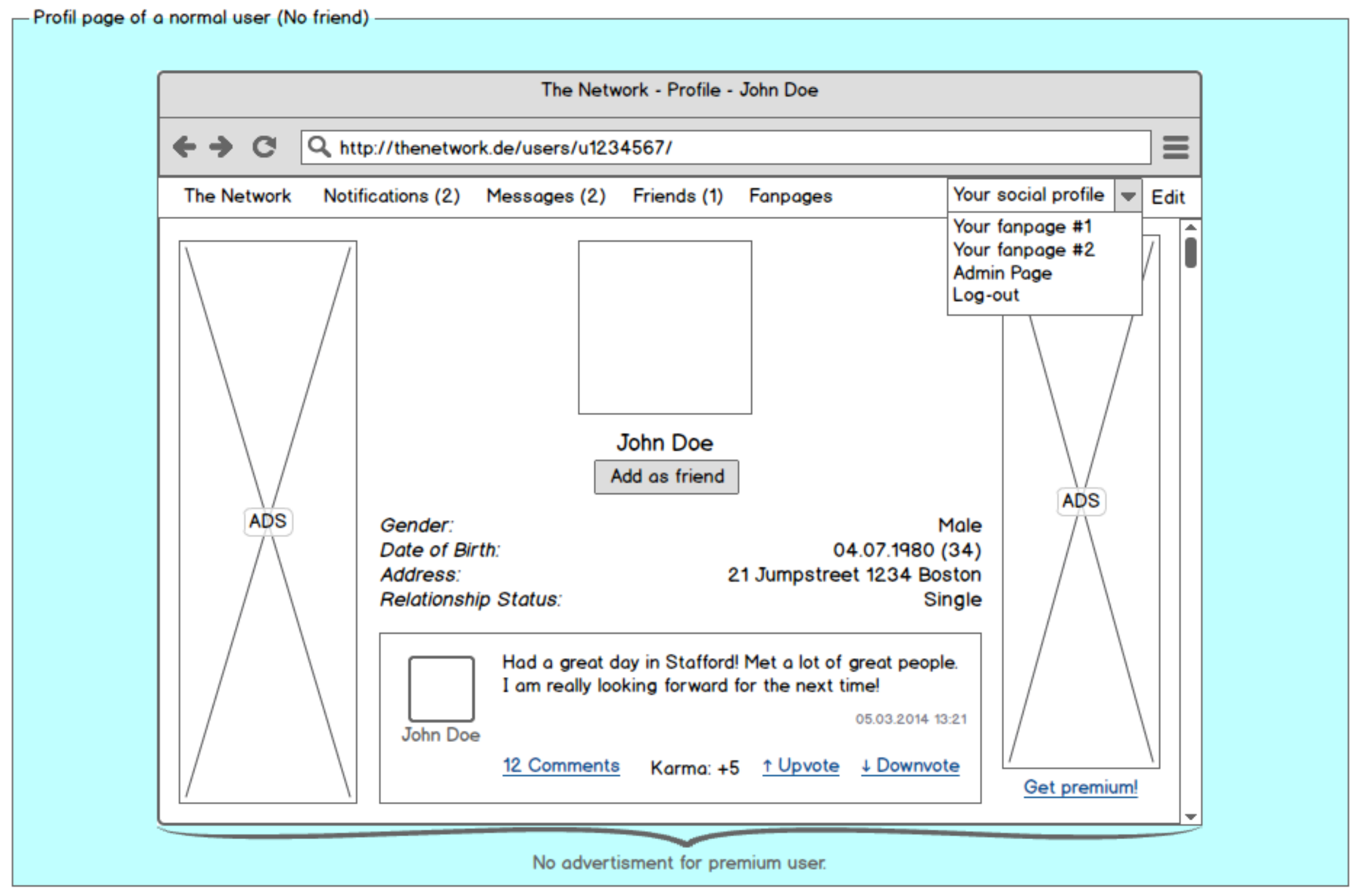
Connected profiles:
Admin page a1234567 Remove
Fanpage f1234567 Remove

Had a great day in Stafford! Met a lot of great people. I am really looking forward for the next time!
You 05.03.2014 13:21
Edit 12 Comments Karma: +5 ↑ Upvote ↓ Downvote

ADS ADS Get premium!

No advertisment for premium user.

Profil page of a normal user (No friend)



The Network - Profile - John Doe

http://thenetwork.de/users/u1234567/

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

John Doe
Add as friend

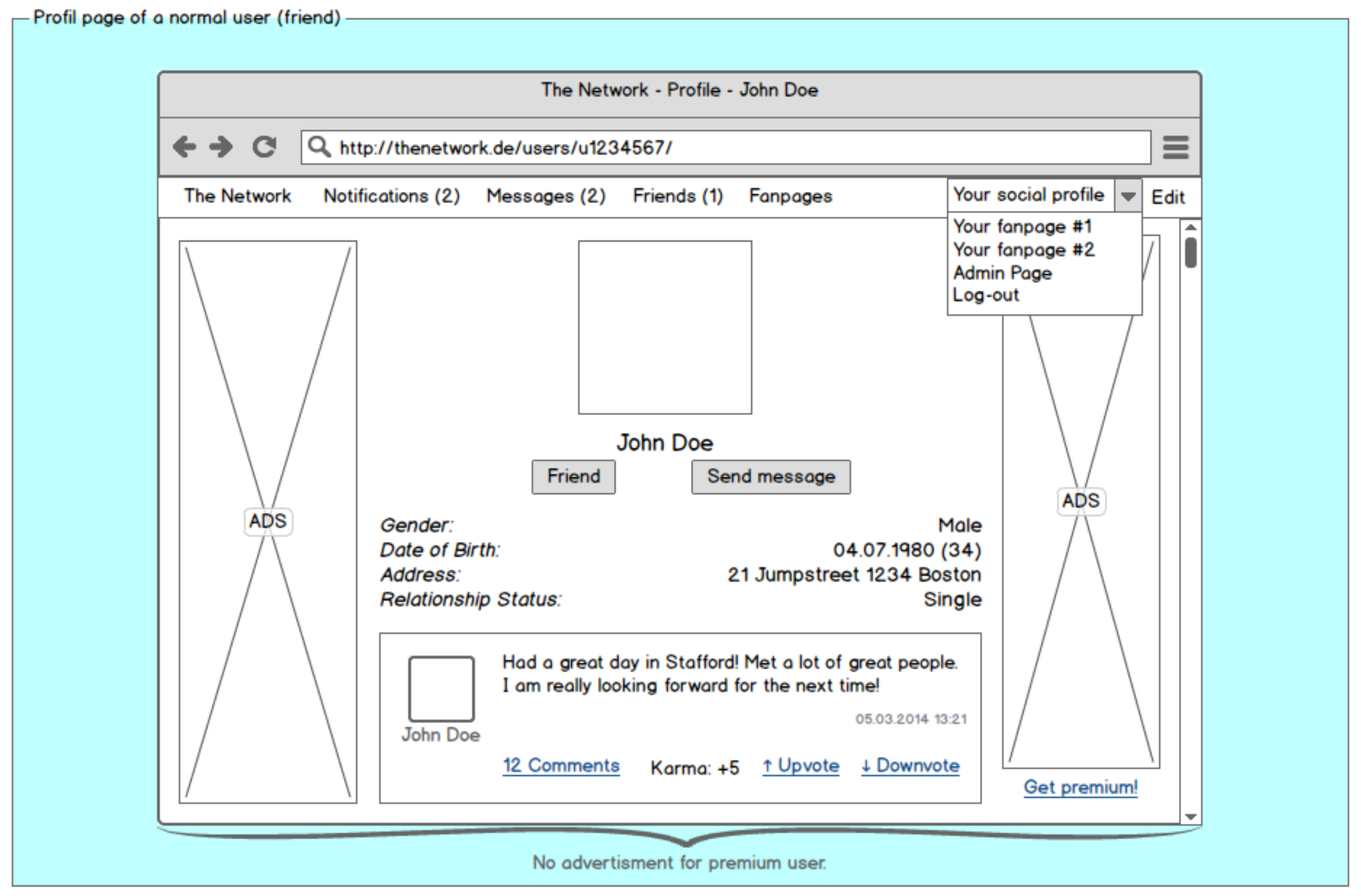
Gender: Male
Date of Birth: 04.07.1980 (34)
Address: 21 Jumpstreet 1234 Boston
Relationship Status: Single

Had a great day in Stafford! Met a lot of great people. I am really looking forward for the next time!
John Doe 05.03.2014 13:21
12 Comments Karma: +5 ↑ Upvote ↓ Downvote

ADS ADS Get premium!

No advertisment for premium user.

Profil page of a normal user (friend)



The Network - Profile - John Doe

http://thenetwork.de/users/u1234567/

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

John Doe
Friend Send message

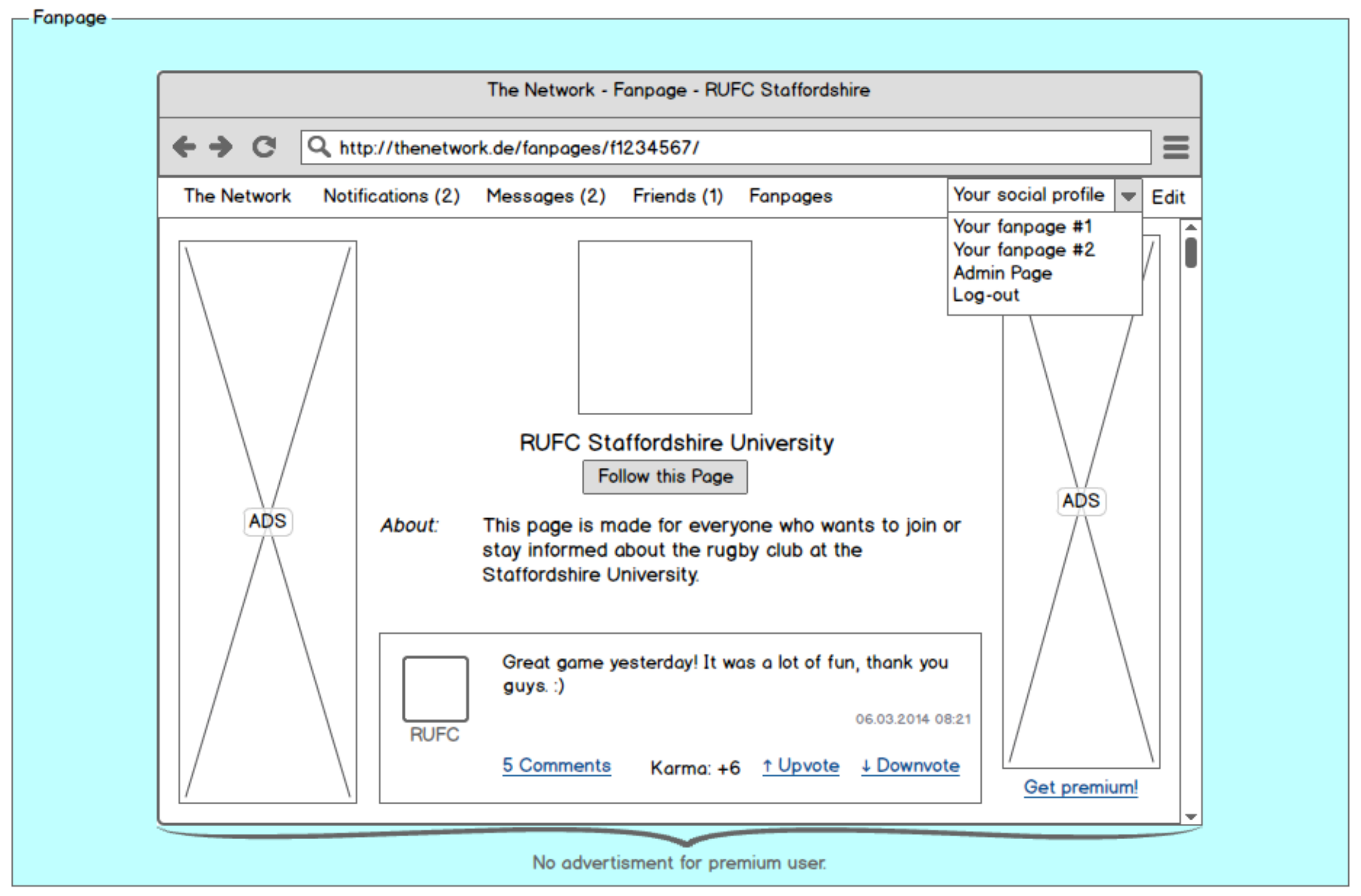
Gender: Male
Date of Birth: 04.07.1980 (34)
Address: 21 Jumpstreet 1234 Boston
Relationship Status: Single

Had a great day in Stafford! Met a lot of great people. I am really looking forward for the next time!
John Doe 05.03.2014 13:21
12 Comments Karma: +5 ↑ Upvote ↓ Downvote

ADS ADS Get premium!

No advertisment for premium user.

Fanpage



The Network - Fanpage - RUFC Staffordshire

http://thenetwork.de/fanpages/f1234567/

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

RUFC Staffordshire University
Follow this Page

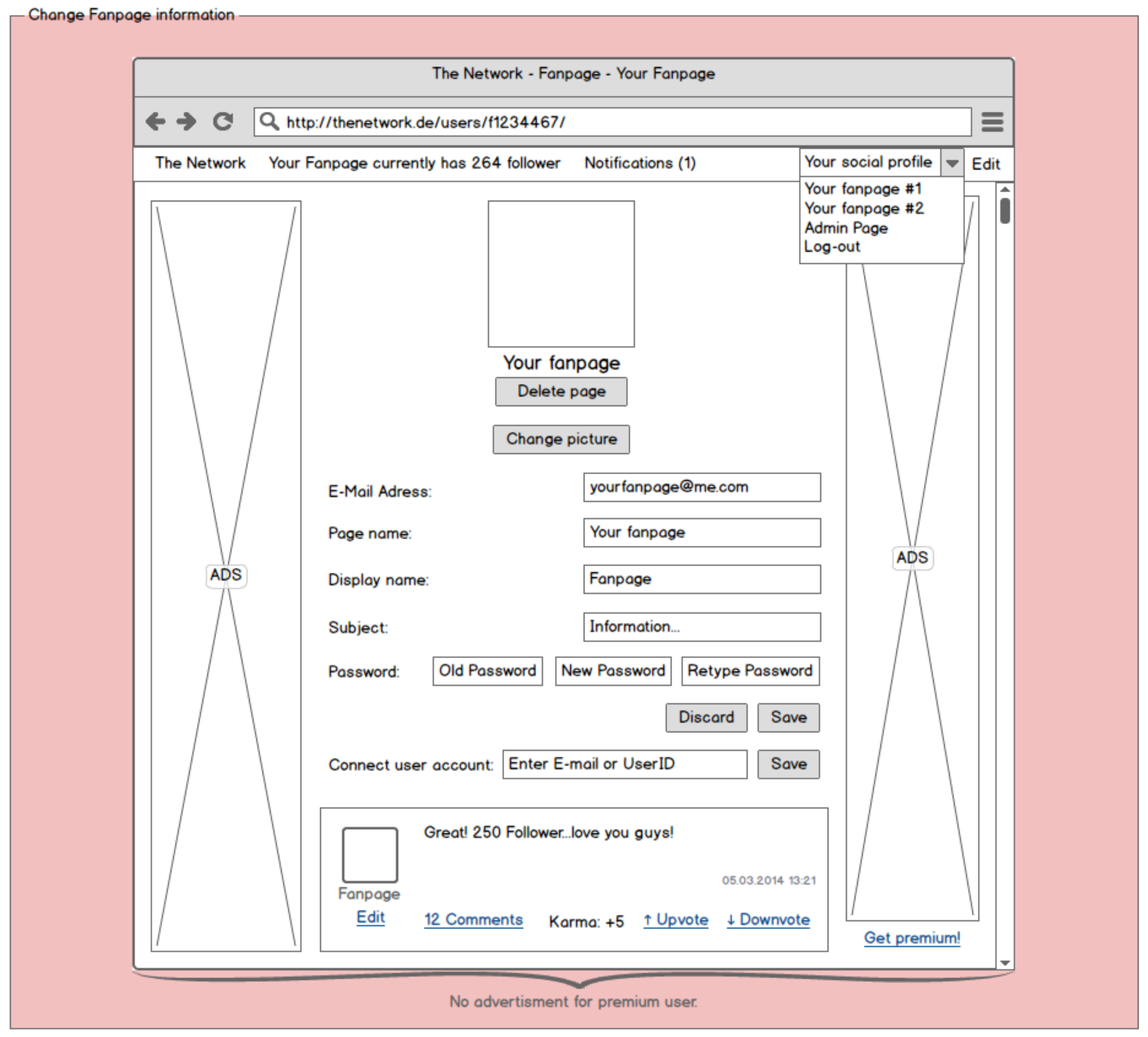
About: This page is made for everyone who wants to join or stay informed about the rugby club at the Staffordshire University.

Great game yesterday! It was a lot of fun, thank you guys. :)
RUFC 06.03.2014 08:21
5 Comments Karma: +6 ↑ Upvote ↓ Downvote

ADS ADS Get premium!

No advertisment for premium user.

Change Fanpage information



The Network - Fanpage - Your Fanpage

http://thenetwork.de/users/f1234467/

The Network Your Fanpage currently has 264 follower Notifications (1) Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

Your fanpage
Delete page
Change picture

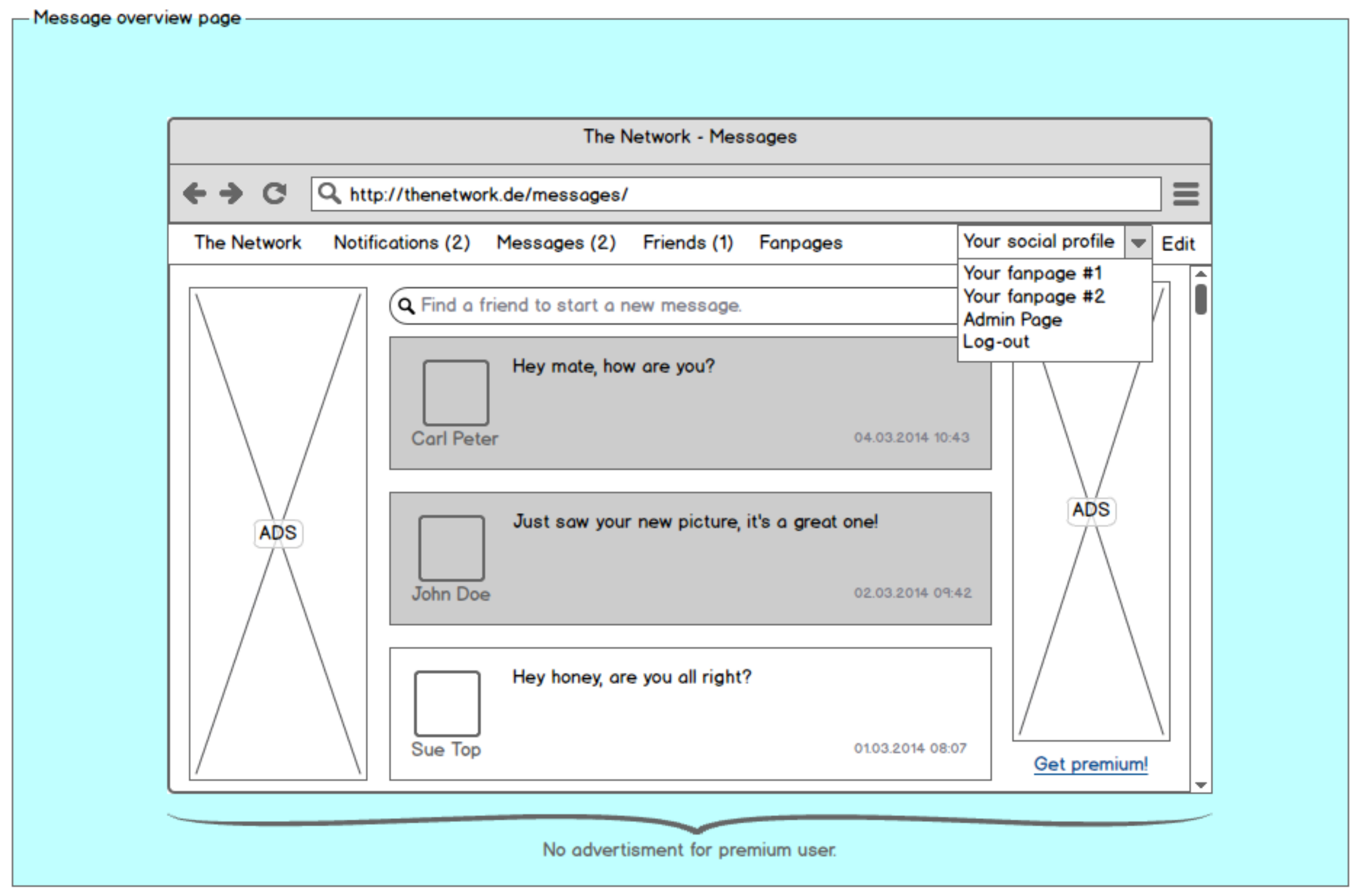
E-Mail Adress: yourfanpage@me.com
Page name: Your fanpage
Display name: Fanpage
Subject: Information...
Password: Old Password New Password Retype Password
Discard Save
Connect user account: Enter E-mail or UserID Save

Great! 250 Follower...love you guys!
Fanpage 05.03.2014 13:21
Edit 12 Comments Karma: +5 ↑ Upvote ↓ Downvote

ADS ADS Get premium!

No advertisment for premium user.

Message overview page



The Network - Messages

http://thenetwork.de/messages/

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

Find a friend to start a new message.

Hey mate, how are you?
Carl Peter 04.03.2014 10:43

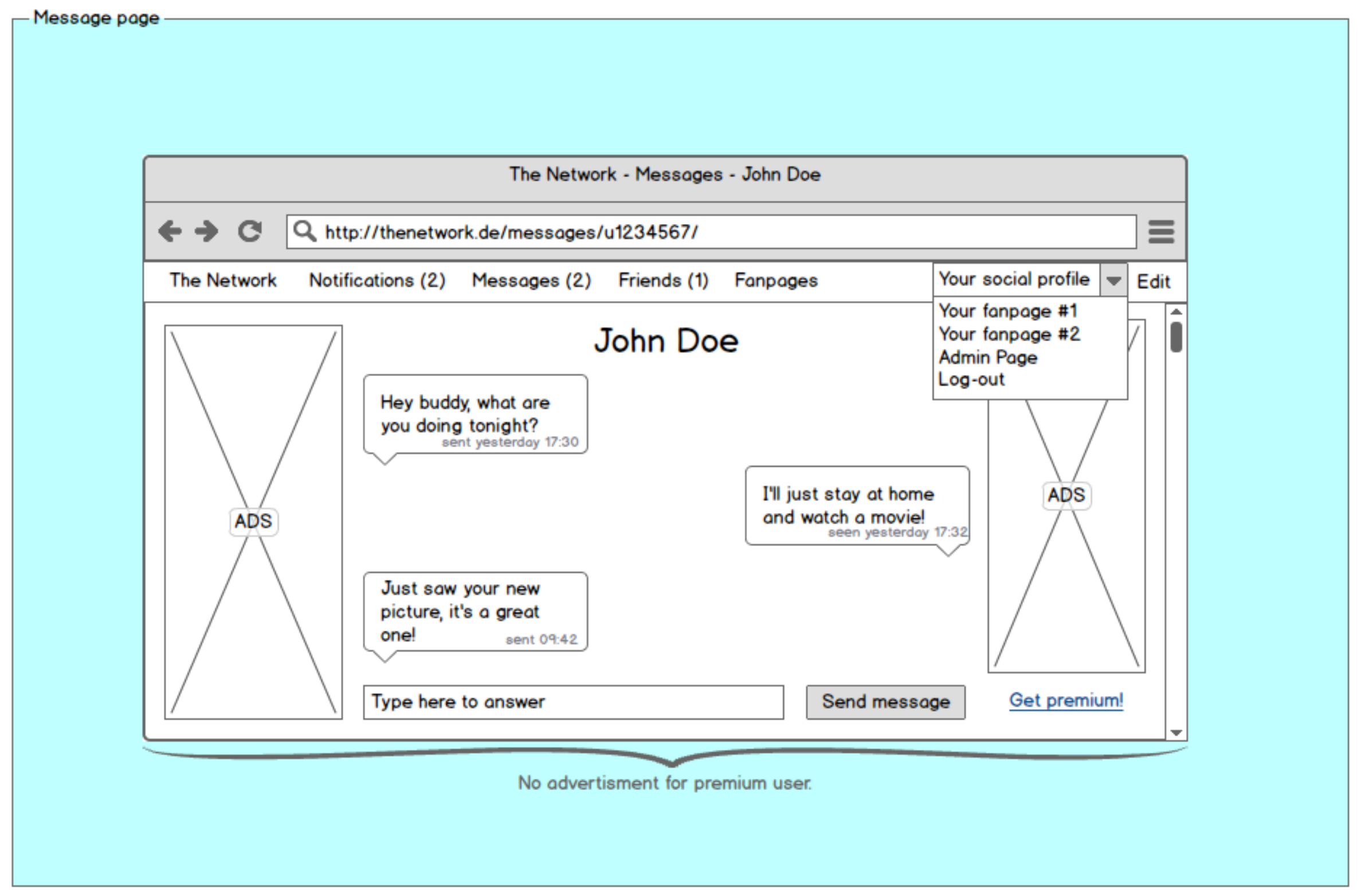
Just saw your new picture, it's a great one!
John Doe 02.03.2014 09:42

Hey honey, are you all right?
Sue Top 01.03.2014 08:07

ADS ADS Get premium!

No advertisment for premium user.

Message page



The Network - Messages - John Doe

http://thenetwork.de/messages/u1234567/

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

John Doe

Hey buddy, what are you doing tonight? sent yesterday 17:30

I'll just stay at home and watch a movie! seen yesterday 17:32

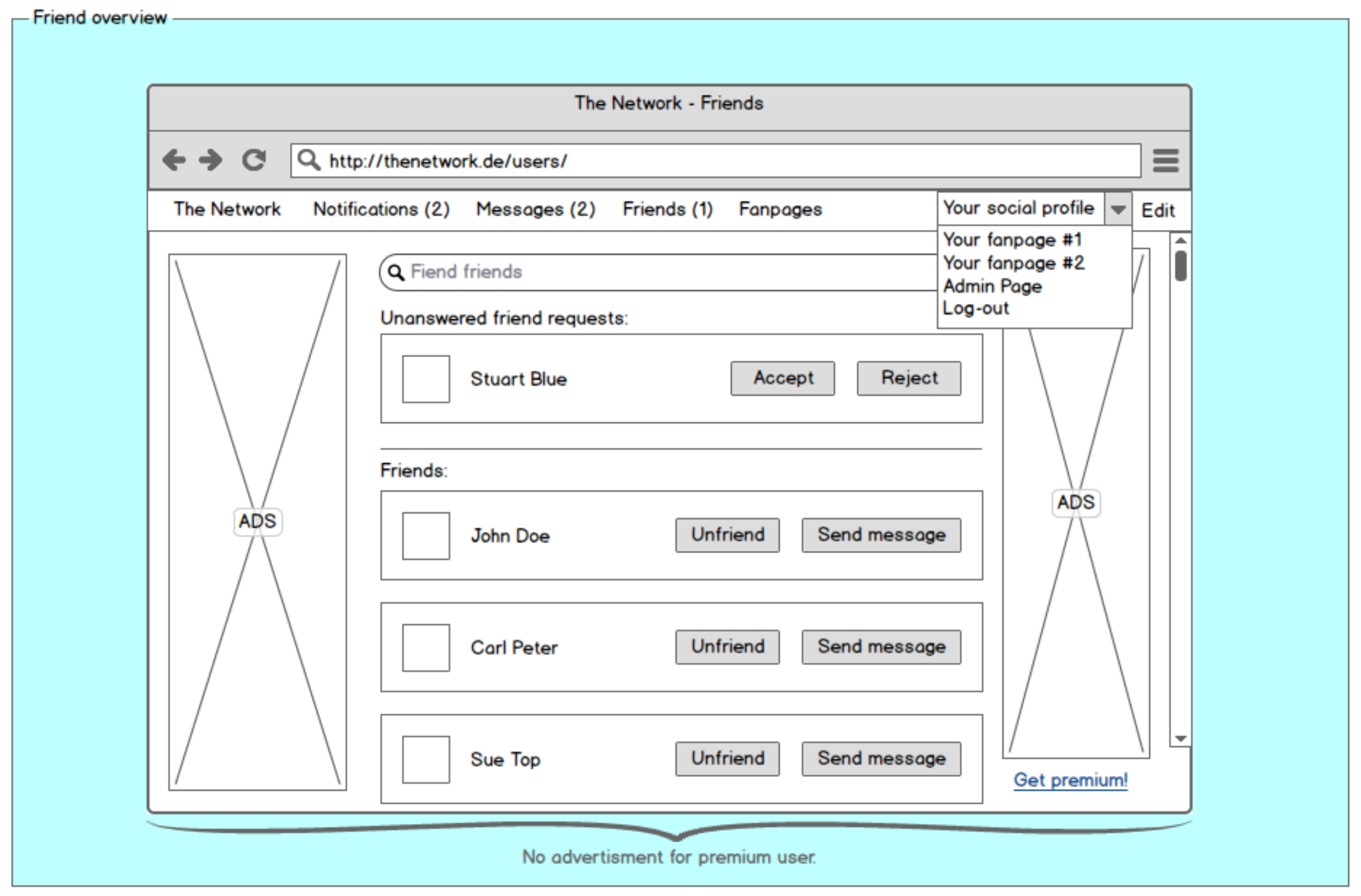
Just saw your new picture, it's a great one! sent 09:42

Type here to answer Send message

ADS ADS Get premium!

No advertisment for premium user.

Friend overview



The Network - Friends

http://thenetwork.de/users/

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

Fiend friends

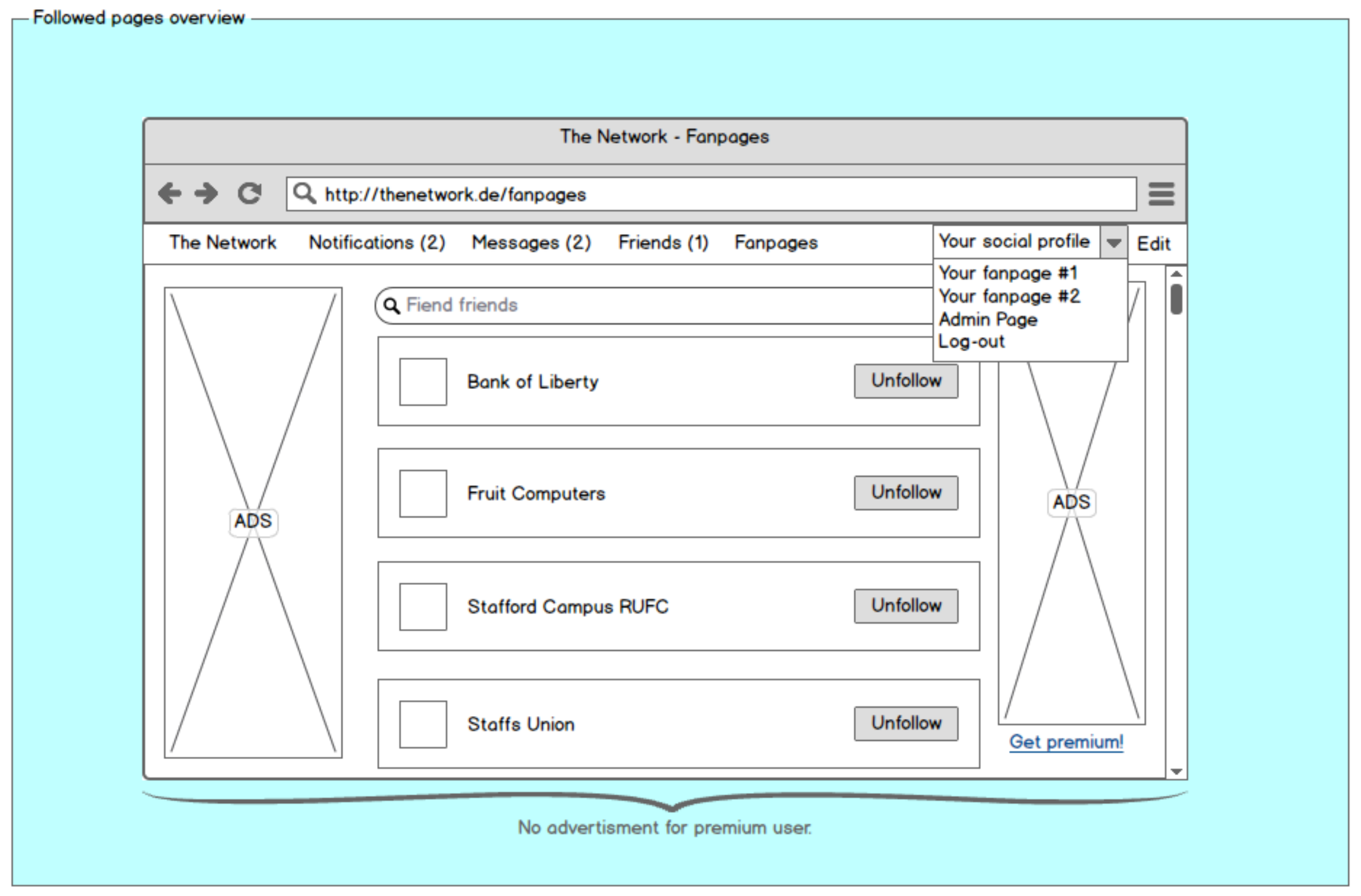
Unanswered friend requests:
Stuart Blue Accept Reject

Friends:
John Doe Unfriend Send message
Carl Peter Unfriend Send message
Sue Top Unfriend Send message

ADS ADS Get premium!

No advertisment for premium user.

Followed pages overview



The Network - Fanpages

http://thenetwork.de/fanpages

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

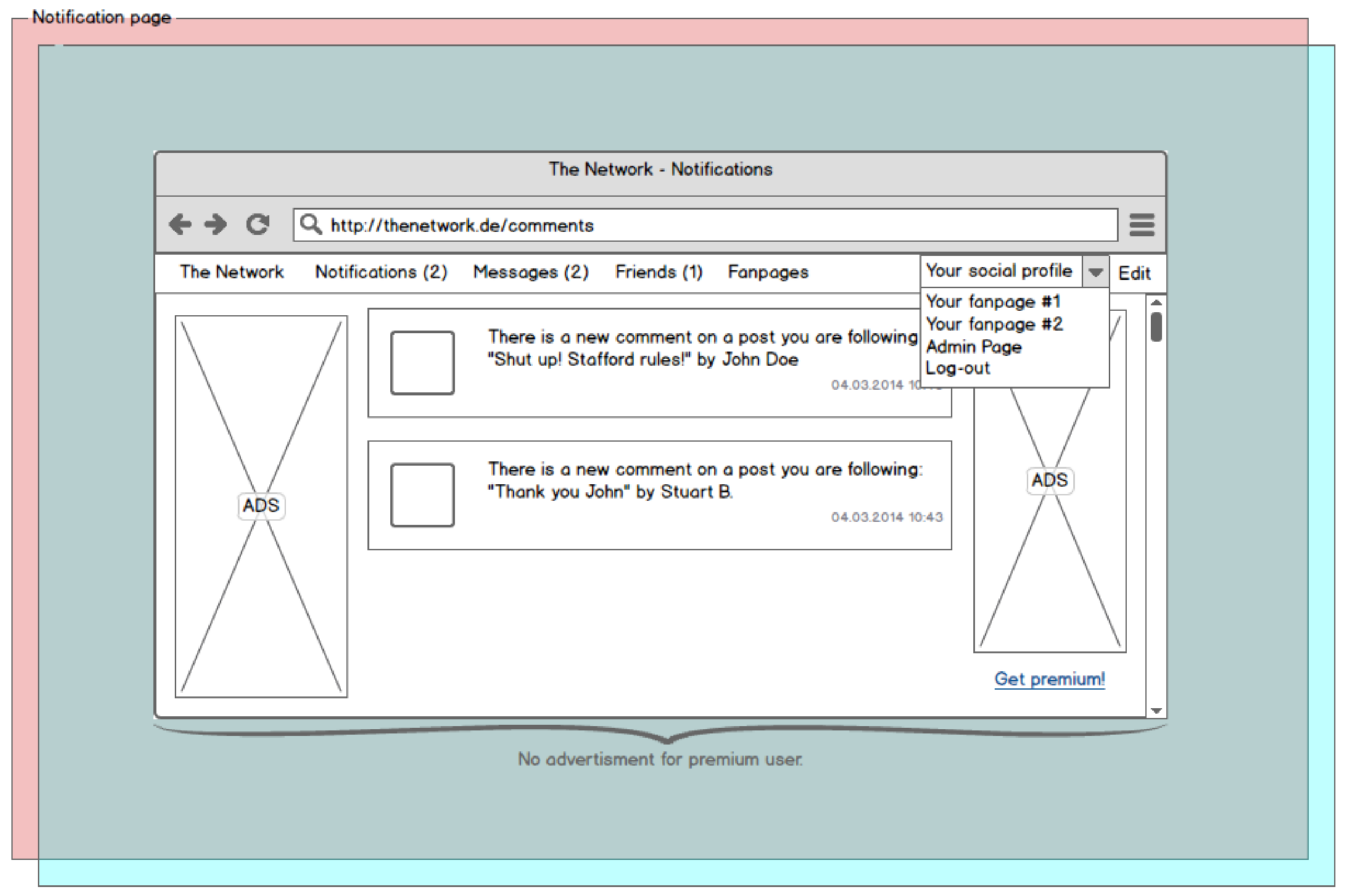
Fiend friends

Bank of Liberty Unfollow
Fruit Computers Unfollow
Stafford Campus RUFC Unfollow
Staffs Union Unfollow

ADS ADS Get premium!

No advertisment for premium user.

Notification page



The Network - Notifications

http://thenetwork.de/comments

The Network Notifications (2) Messages (2) Friends (1) Fanpages Your social profile Edit

Your fanpage #1
Your fanpage #2
Admin Page
Log-out

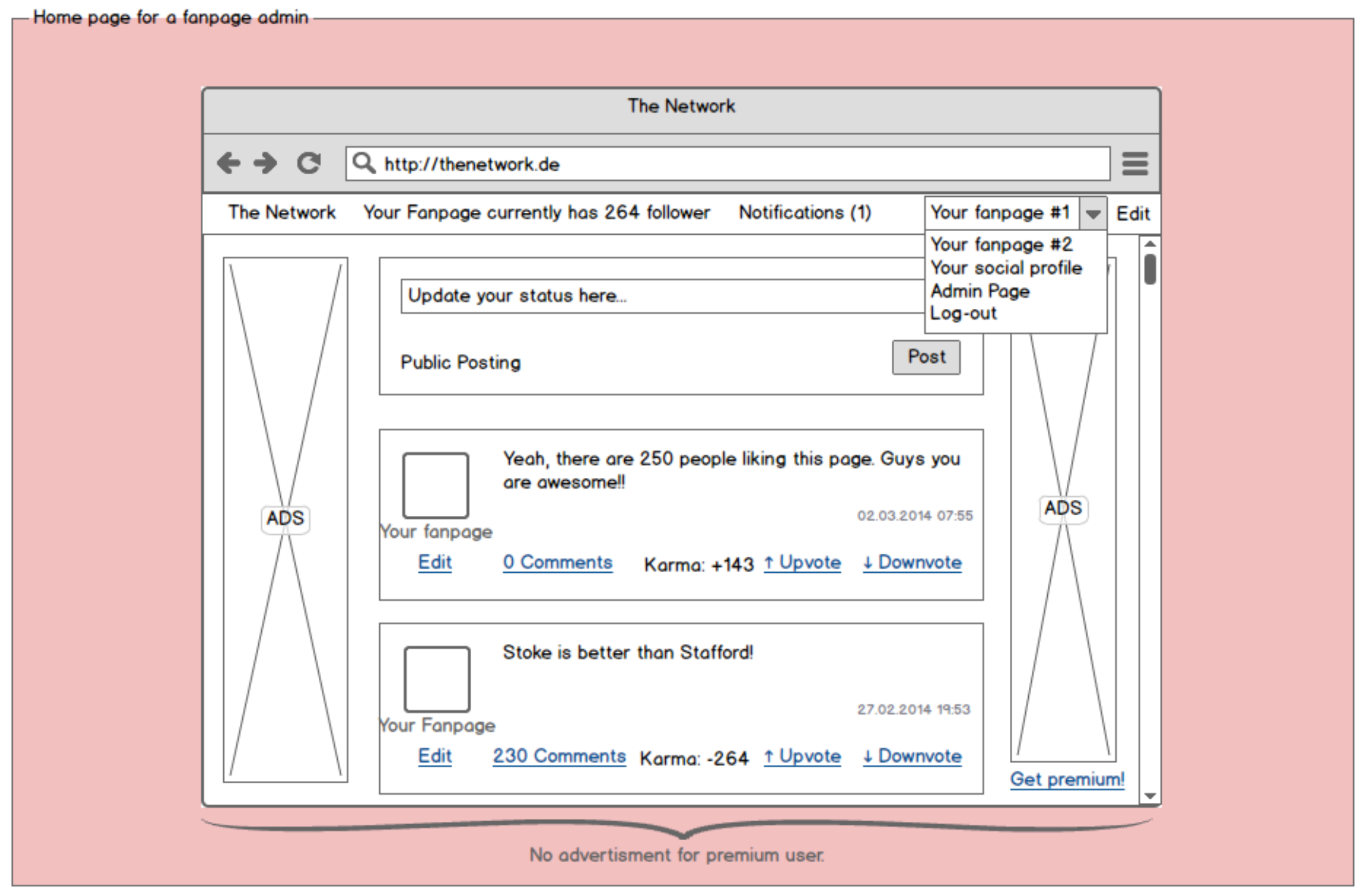
There is a new comment on a post you are following: "Shut up! Stafford rules!" by John Doe
04.03.2014 10:43

There is a new comment on a post you are following: "Thank you John" by Stuart B.
04.03.2014 10:43

ADS ADS Get premium!

No advertisment for premium user.

Home page for a fanpage admin



The Network

http://thenetwork.de

The Network Your Fanpage currently has 264 follower Notifications (1) Your fanpage #1 Edit

Your fanpage #2
Your social profile
Admin Page
Log-out

Update your status here...
Public Posting Post

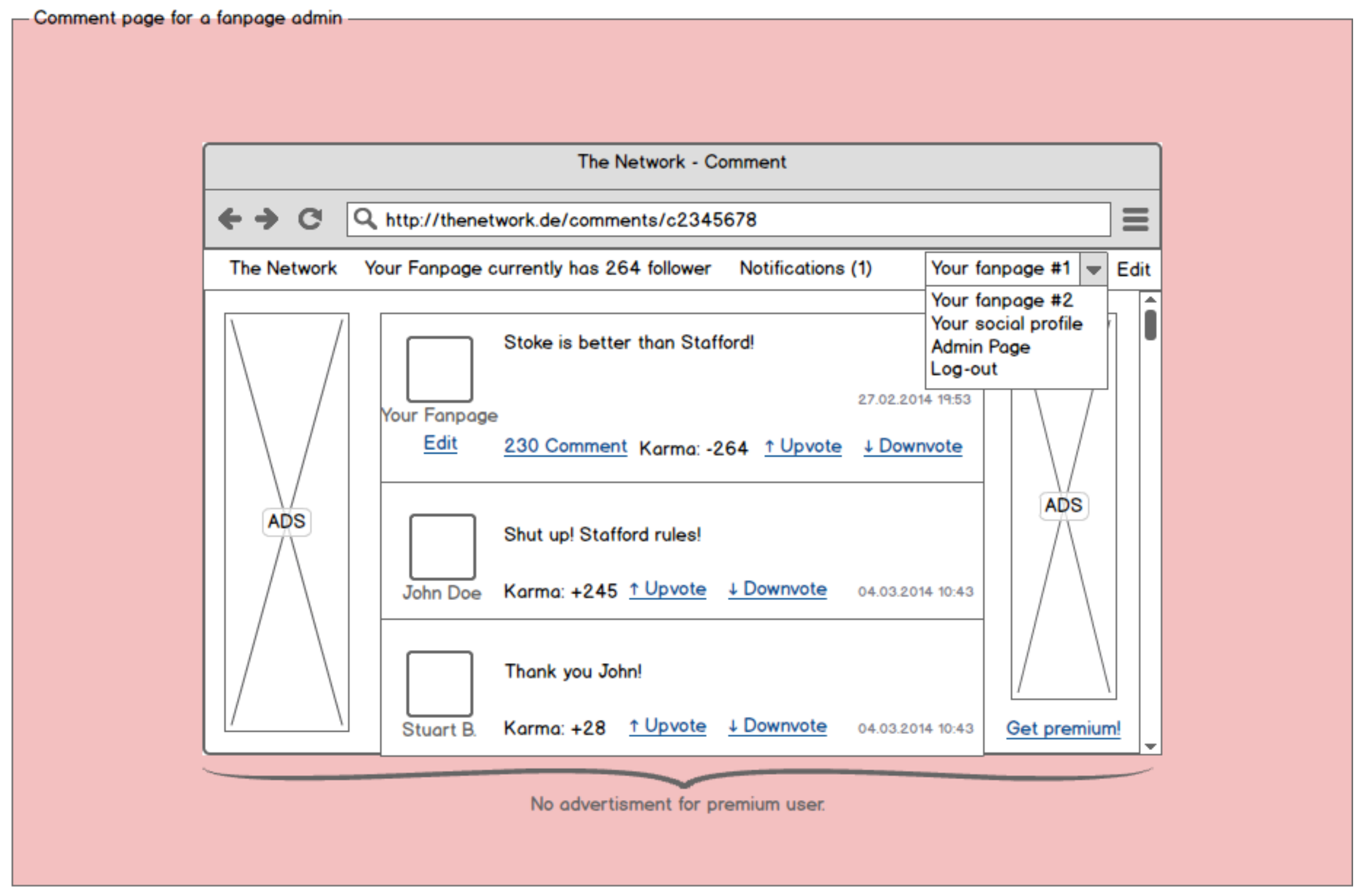
Yeah, there are 250 people liking this page. Guys you are awesome!!
Your fanpage 02.03.2014 07:55
Edit 0 Comments Karma: +143 ↑ Upvote ↓ Downvote

Stoke is better than Stafford!
Your Fanpage 27.02.2014 19:53
Edit 230 Comments Karma: -264 ↑ Upvote ↓ Downvote

ADS ADS Get premium!

No advertisment for premium user.

Comment page for a fanpage admin



The Network - Comment

http://thenetwork.de/comments/c2345678

The Network Your Fanpage currently has 264 follower Notifications (1) Your fanpage #1 Edit

Your fanpage #2
Your social profile
Admin Page
Log-out

Stoke is better than Stafford!
Your Fanpage 27.02.2014 19:53
Edit 230 Comment Karma: -264 ↑ Upvote ↓ Downvote

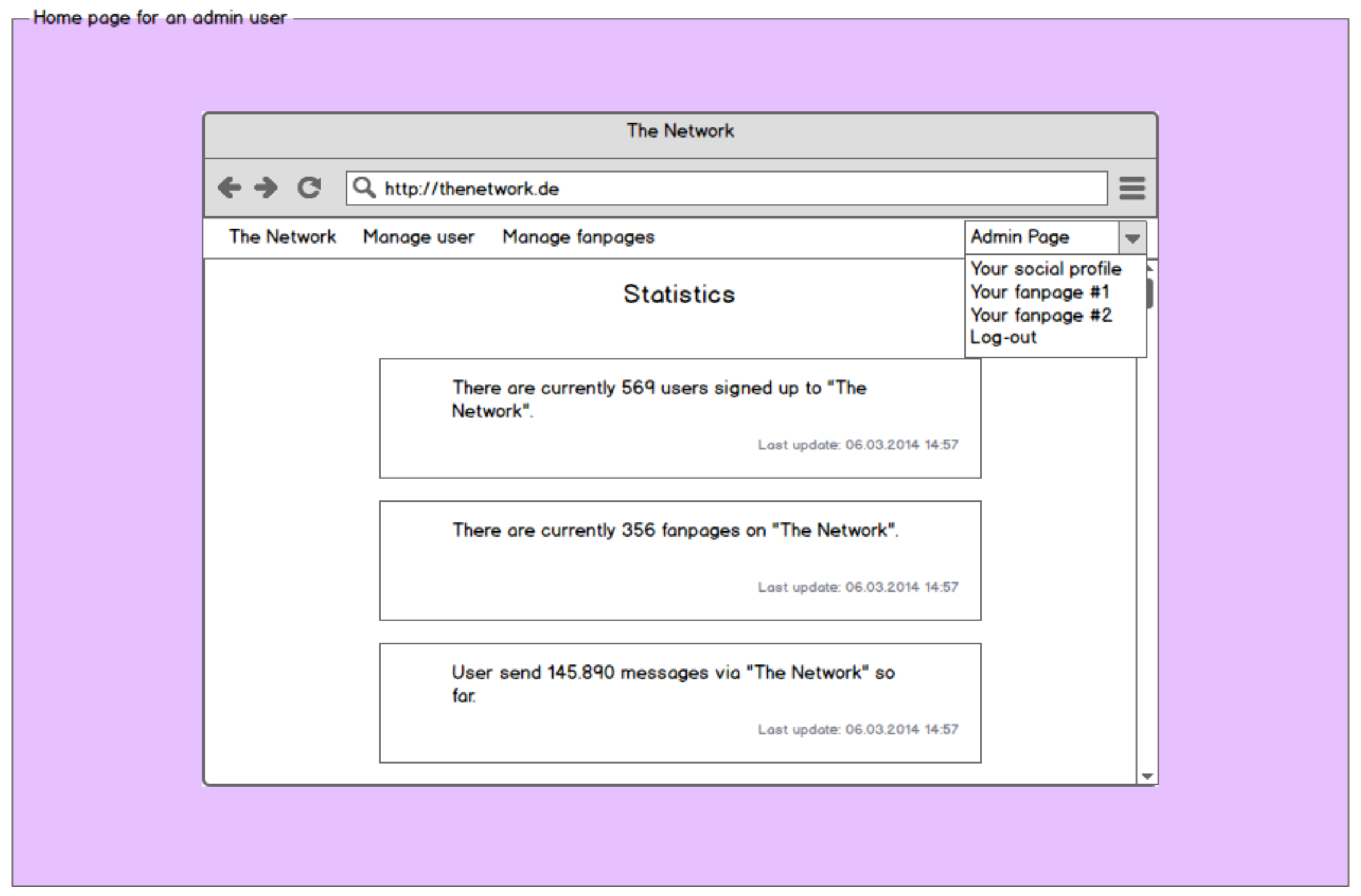
Shut up! Stafford rules!
John Doe Karma: +245 ↑ Upvote ↓ Downvote 04.03.2014 10:43

Thank you John!
Stuart B. Karma: +28 ↑ Upvote ↓ Downvote 04.03.2014 10:43

ADS ADS Get premium!

No advertisment for premium user.

Home page for an admin user



The Network

http://thenetwork.de

The Network Manage user Manage fanpages Admin Page

Your social profile
Your fanpage #1
Your fanpage #2
Log-out

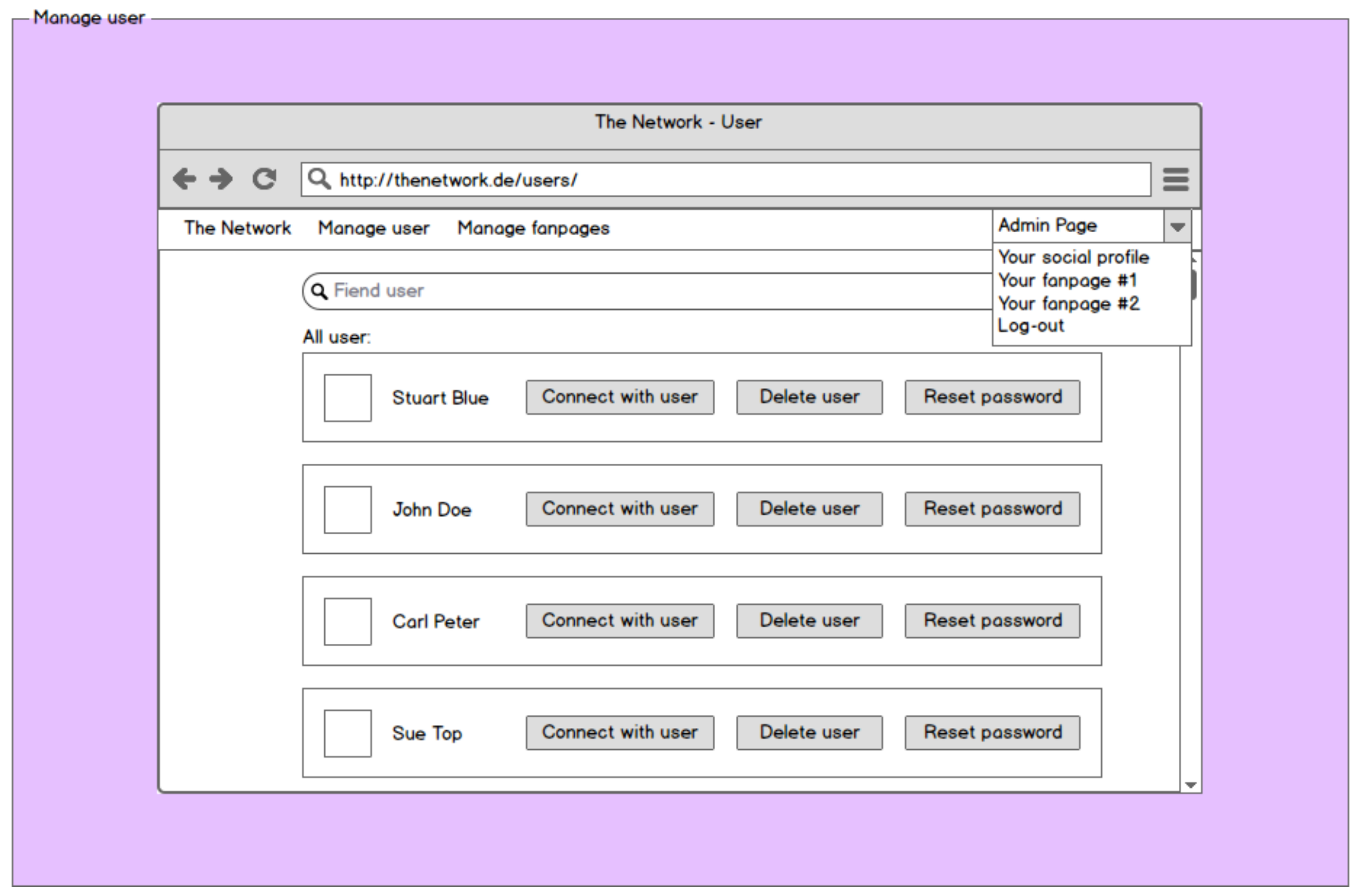
Statistics

There are currently 569 users signed up to "The Network".
Last update: 06.03.2014 14:57

There are currently 356 fanpages on "The Network".
Last update: 06.03.2014 14:57

User send 145.890 messages via "The Network" so far.
Last update: 06.03.2014 14:57

Manage user



The Network - User

http://thenetwork.de/users/

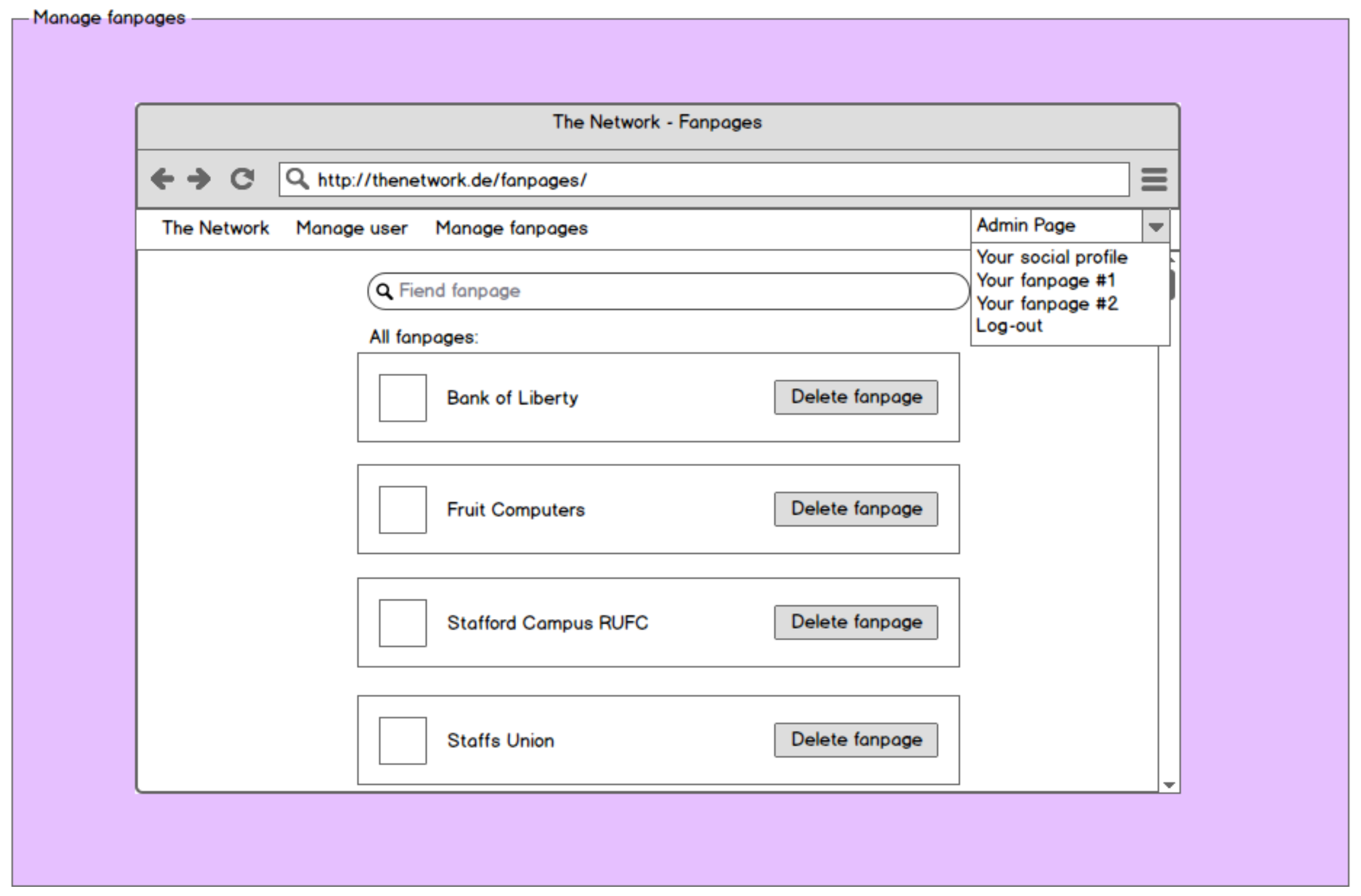
The Network Manage user Manage fanpages Admin Page

Your social profile
Your fanpage #1
Your fanpage #2
Log-out

Fiend user

All user:
Stuart Blue Connect with user Delete user Reset password
John Doe Connect with user Delete user Reset password
Carl Peter Connect with user Delete user Reset password
Sue Top Connect with user Delete user Reset password

Manage fanpages



The Network - Fanpages

http://thenetwork.de/fanpages/

The Network Manage user Manage fanpages Admin Page

Your social profile
Your fanpage #1
Your fanpage #2
Log-out

Fiend fanpage

All fanpages:
Bank of Liberty Delete fanpage
Fruit Computers Delete fanpage
Stafford Campus RUFC Delete fanpage
Staffs Union Delete fanpage

🟦 Accessed by normal user

🟥 Accessed by page admin

🟪 Accessed by admin user

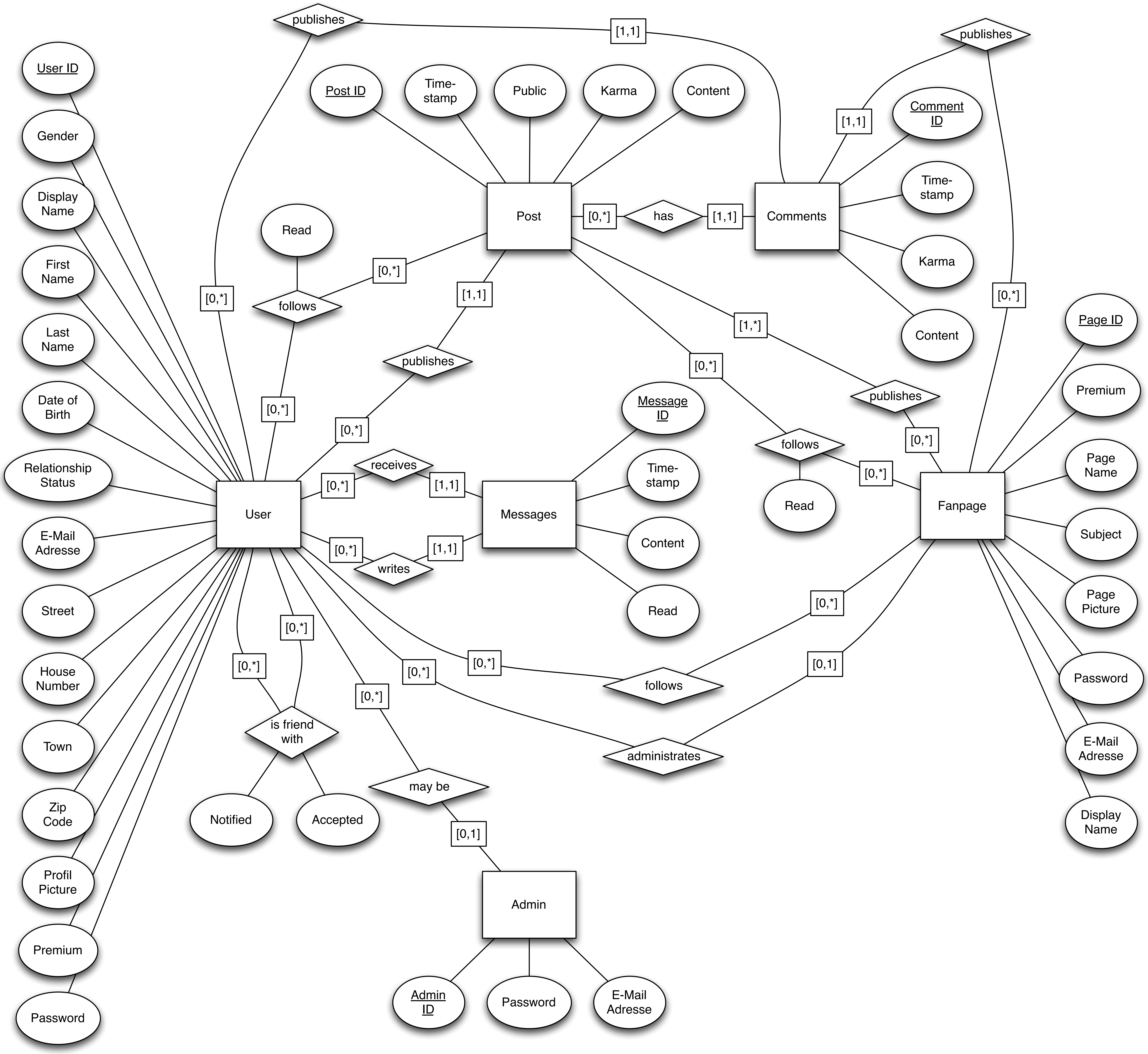
□ Accessed by every user

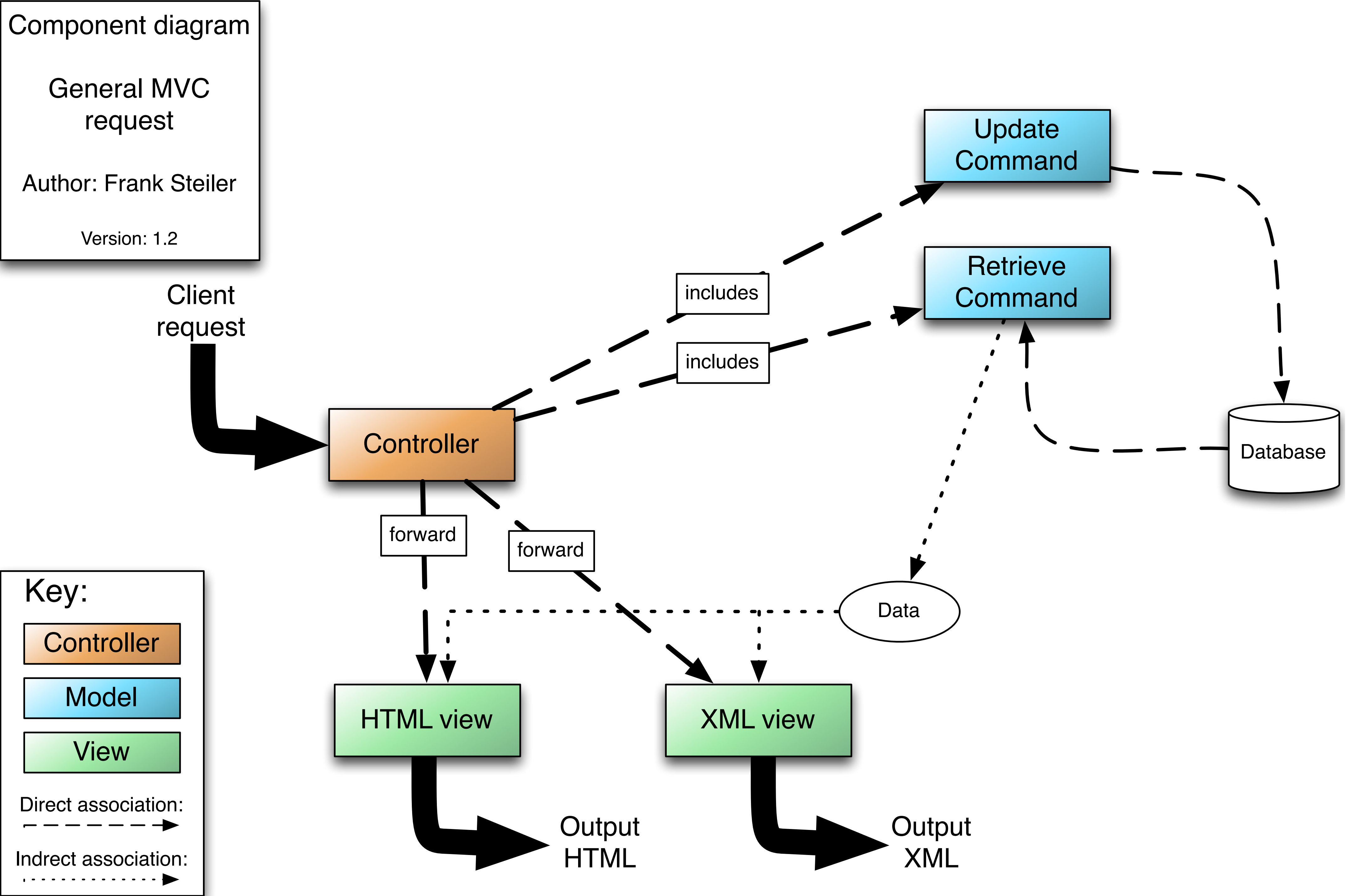
Entity Relationship Diagram

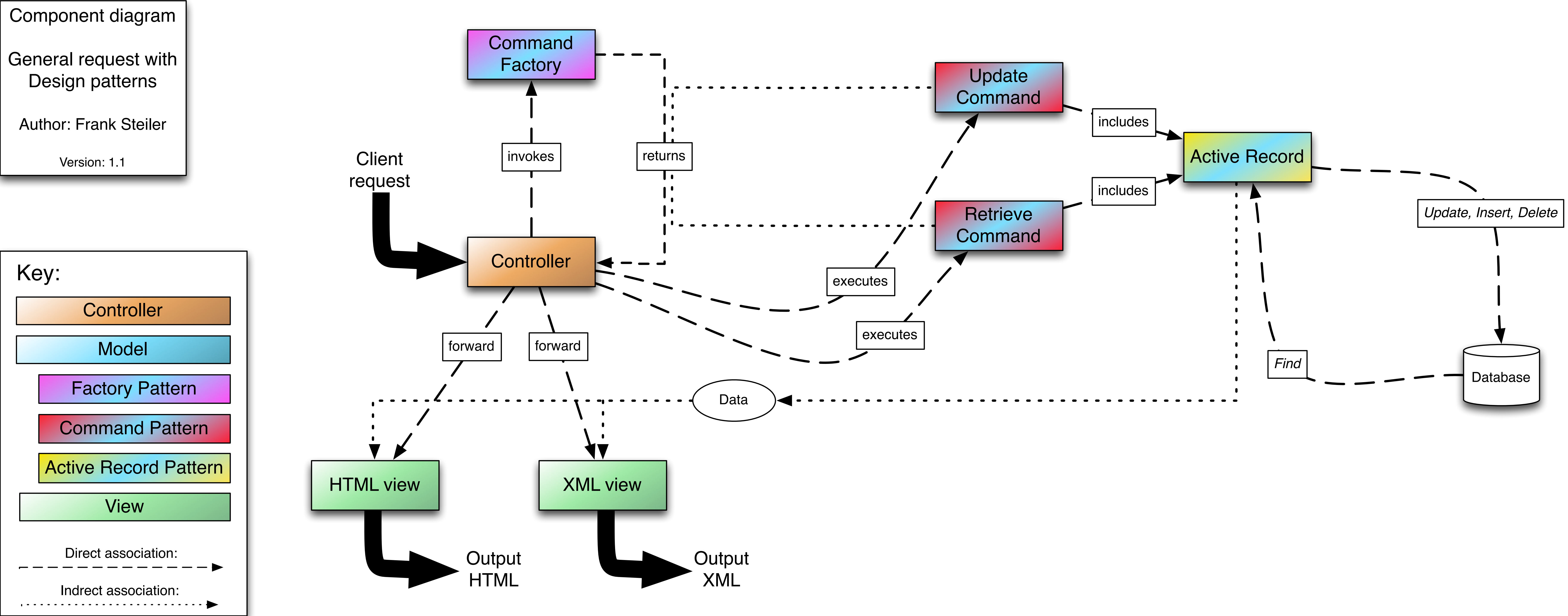
The Network

Author: Frank Steiler

Version: 1.2




Component diagram
General MVC request
Author: Frank Steiler
Version: 1.2
Client request
Controller
includes
includes
Update Command
Retrieve Command
Database
forward
forward
Data
HTML view
XML view
Output HTML
Output XML
Key:
Controller
Model
View
Direct association:
Indrect association:



Component diagram

General request with Design patterns

Author: Frank Steiler

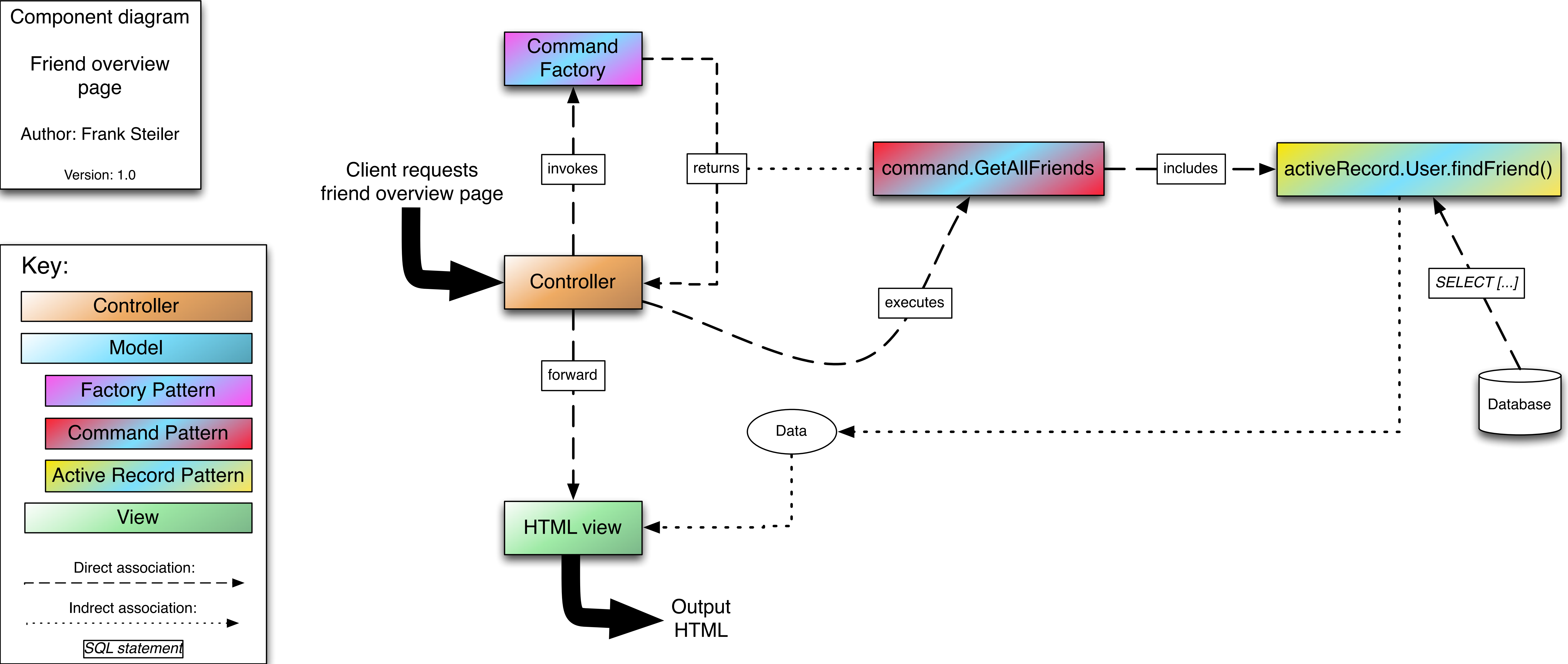
Version: 1.1

Component diagram

Friend overview page

Author: Frank Steiler

Version: 1.0

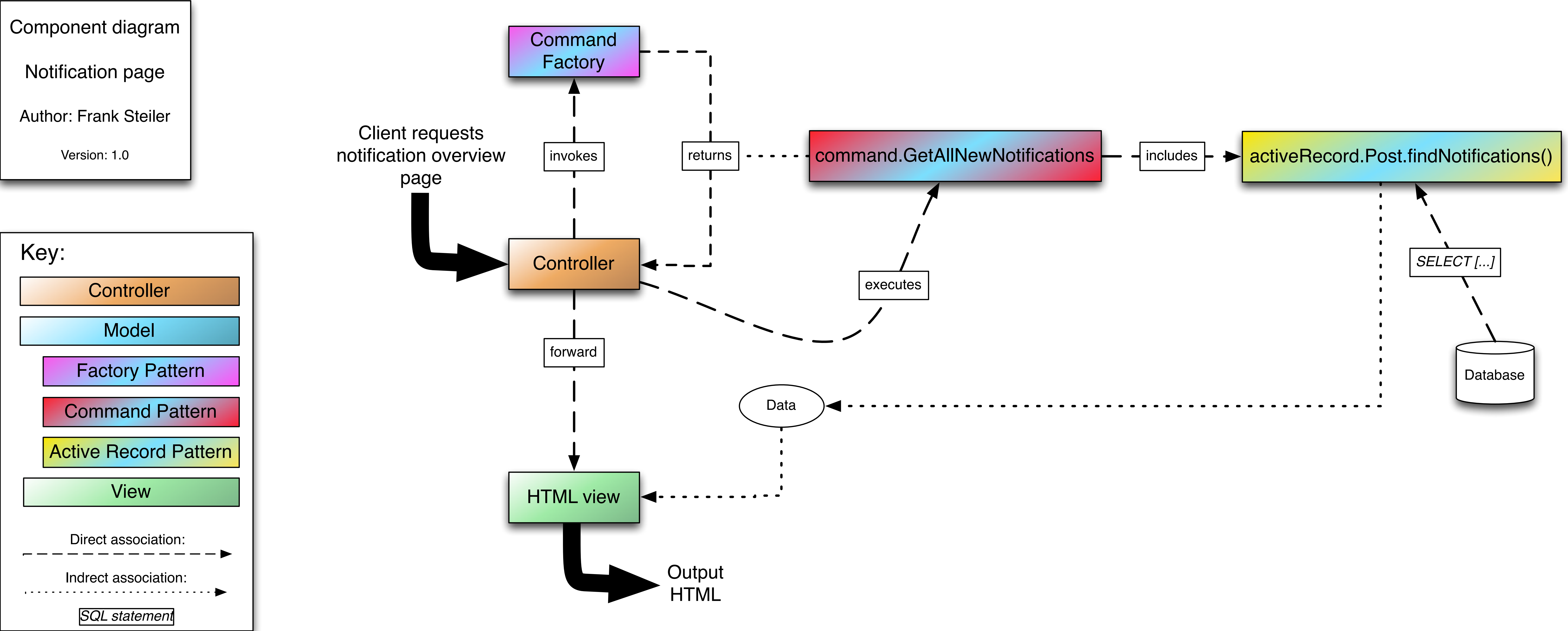


Component diagram

Notification page

Author: Frank Steiler

Version: 1.0

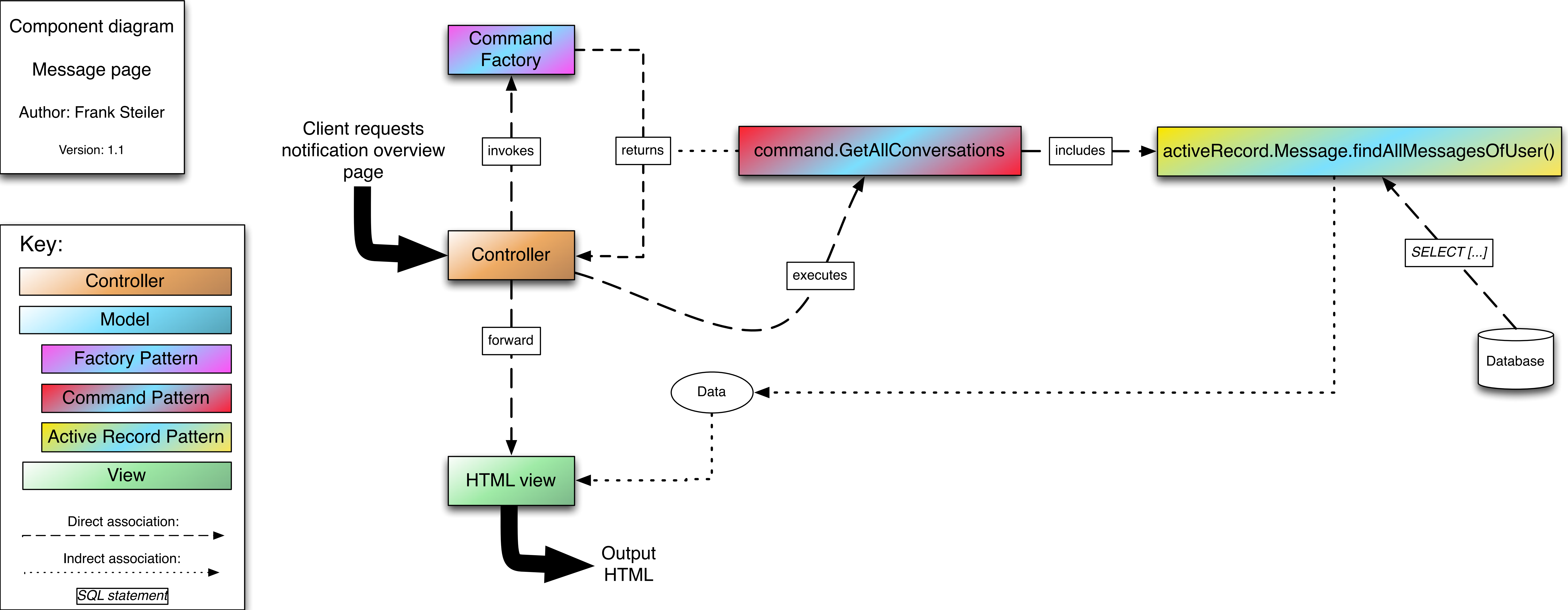


Component diagram

Message page

Author: Frank Steiler

Version: 1.1



Component diagram

User Home Page

Author: Frank Steiler

Version: 1.0

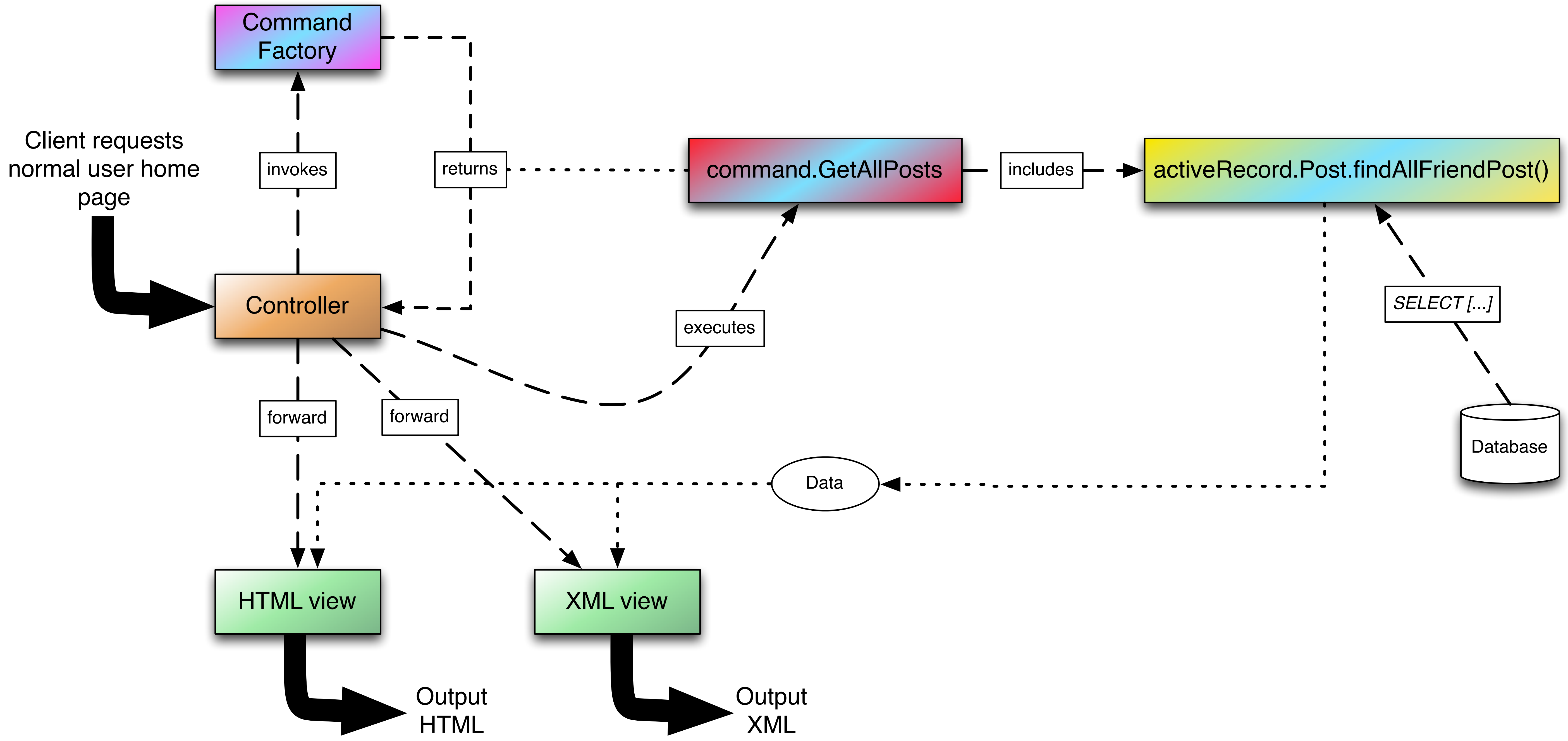
Key:

- Controller
- Model
- Factory Pattern
- Command Pattern
- Active Record Pattern
- View

Direct association: - - - - - - ►

Indrect association: · · · · · · · ►

SQL statement



Client requests normal user home page

Command Factory

invokes

returns

Controller

command.GetAllPosts

includes

activeRecord.Post.findAllFriendPost()

executes

SELECT [...]

Database

forward

forward

Data

HTML view

XML view

Output HTML

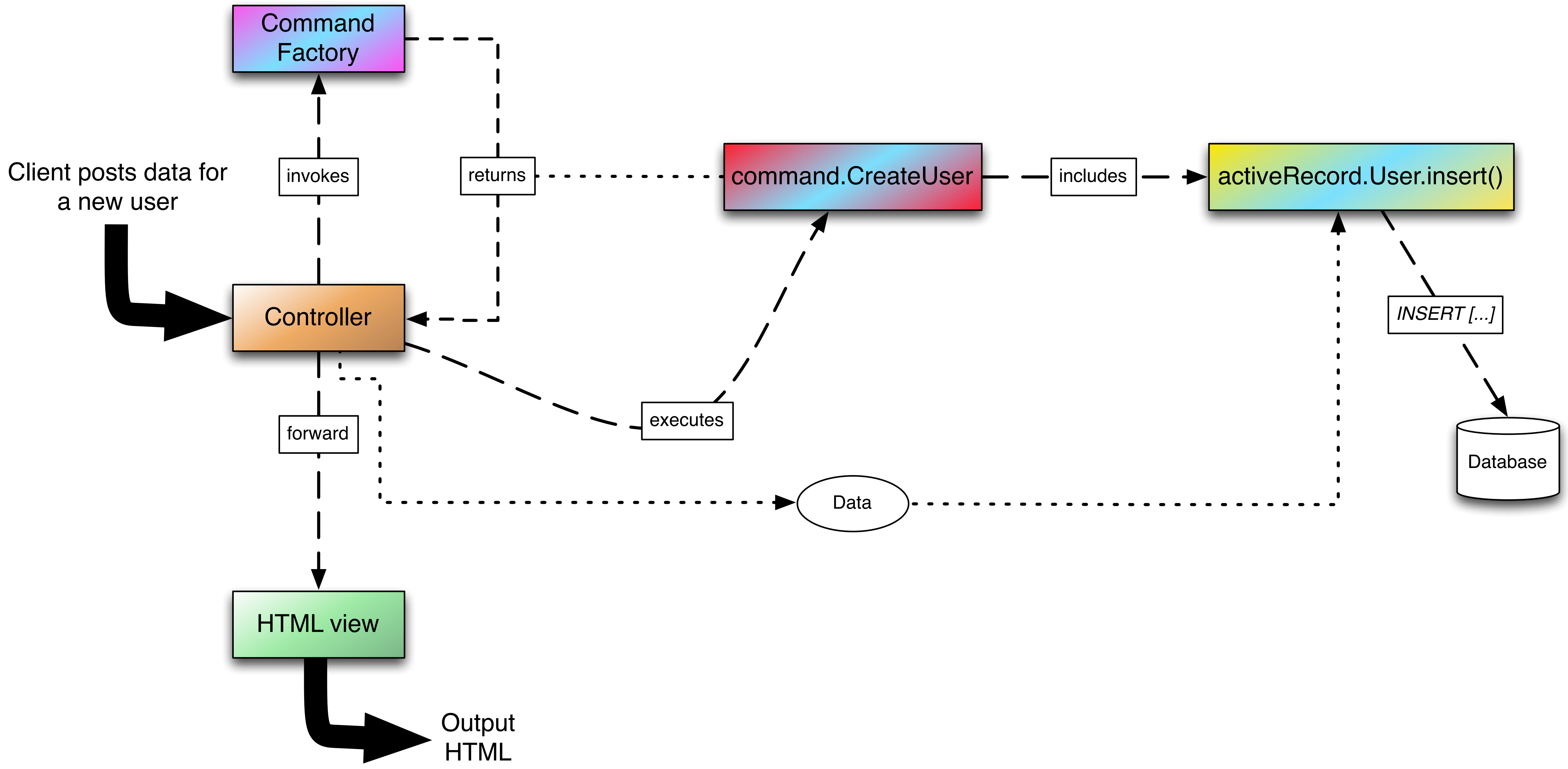
Output XML

Component diagram

Create user

Author: Frank Steiler

Version: 1.0


Client posts data for a new user

Command Factory

invokes

returns

Controller

command.CreateUser

includes

activeRecord.User.insert()

INSERT [...]

Database

executes

forward

Data

HTML view

Output HTML

Key:

- Controller
- Model
- Factory Pattern
- Command Pattern
- Active Record Pattern
- View

Direct association:

Indrect association:

SQL statement

Component diagram

Publish post

Author: Frank Steiler

Version: 1.0

Key:

Controller

Model

Factory Pattern

Command Pattern

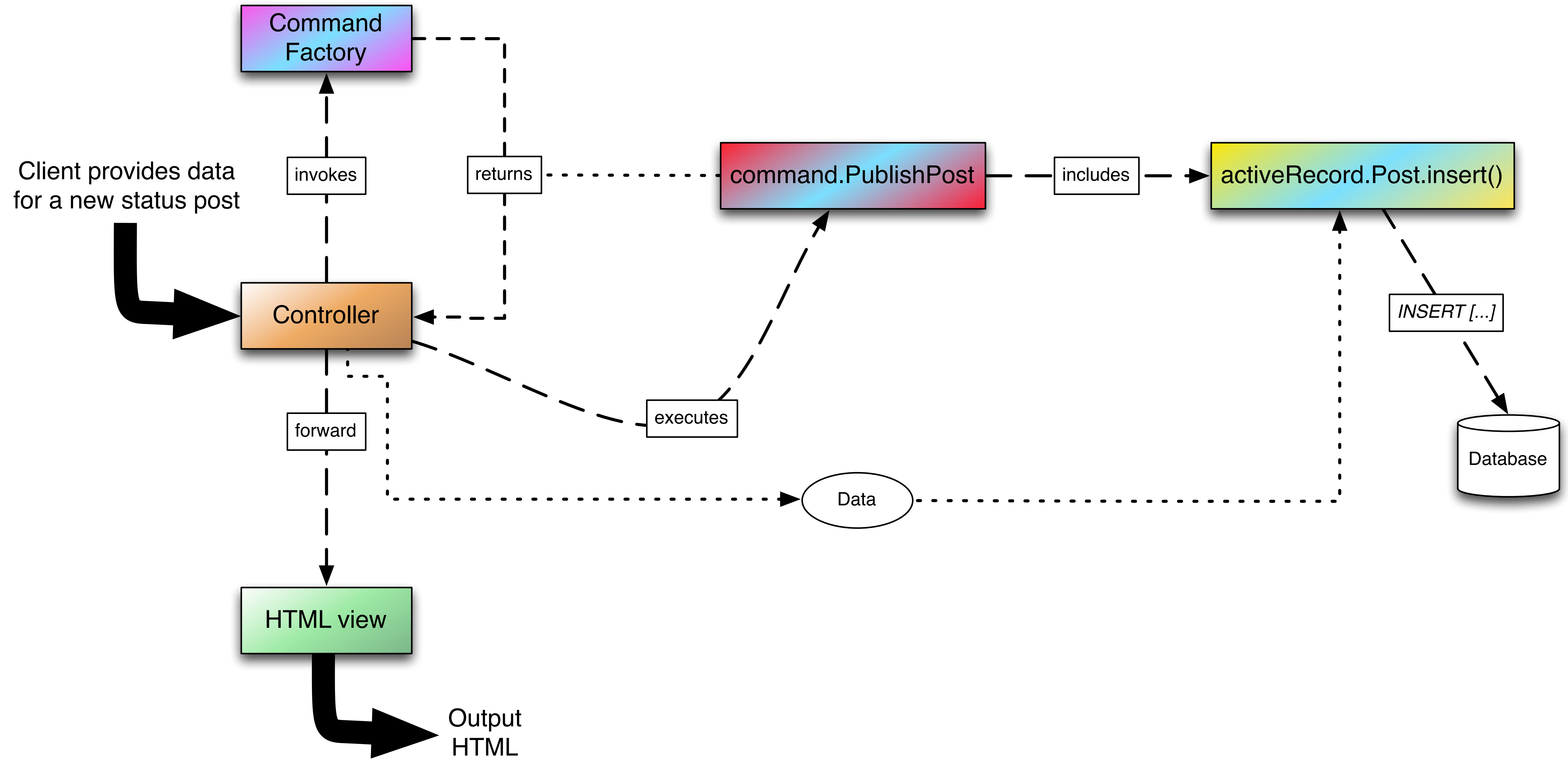
Active Record Pattern

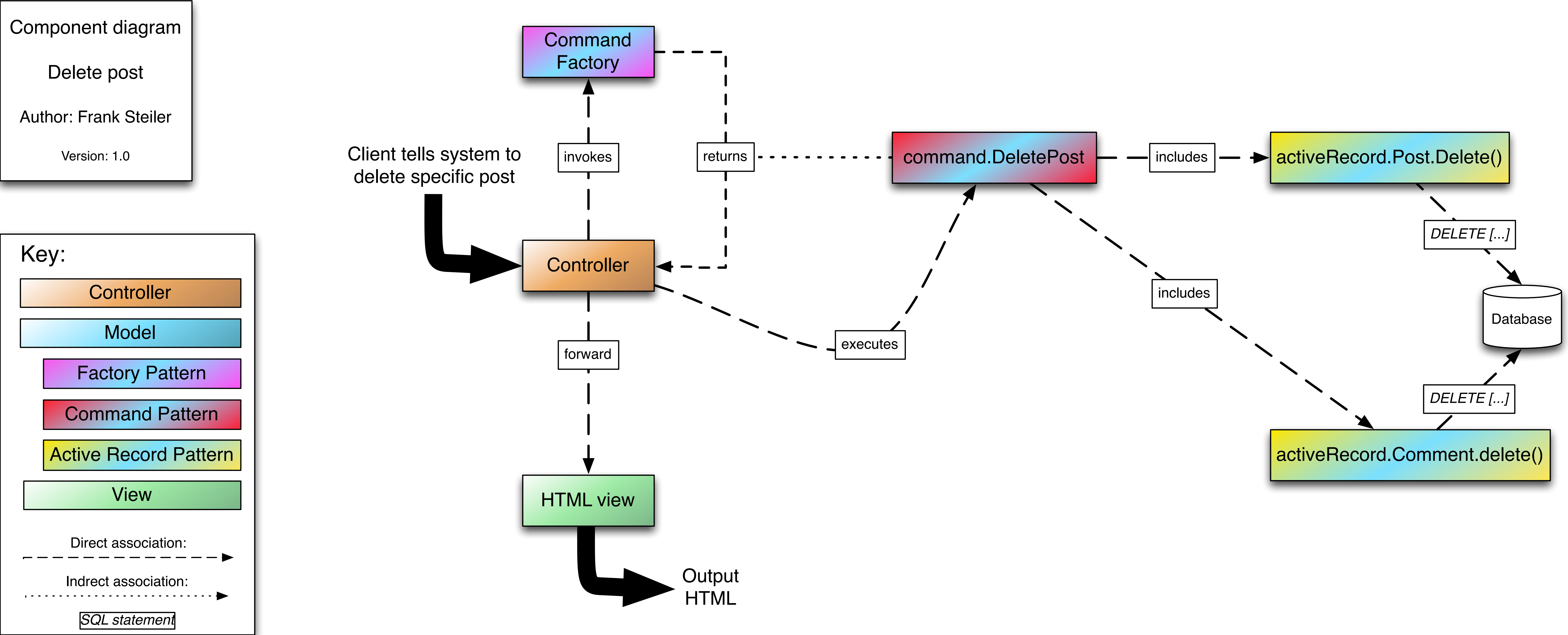
View

Direct association:

Indrect association:

SQL statement





Component diagram

Delete post

Author: Frank Steiler

Version: 1.0

Component diagram

Update Profile Information

Author: Frank Steiler

Version: 1.0

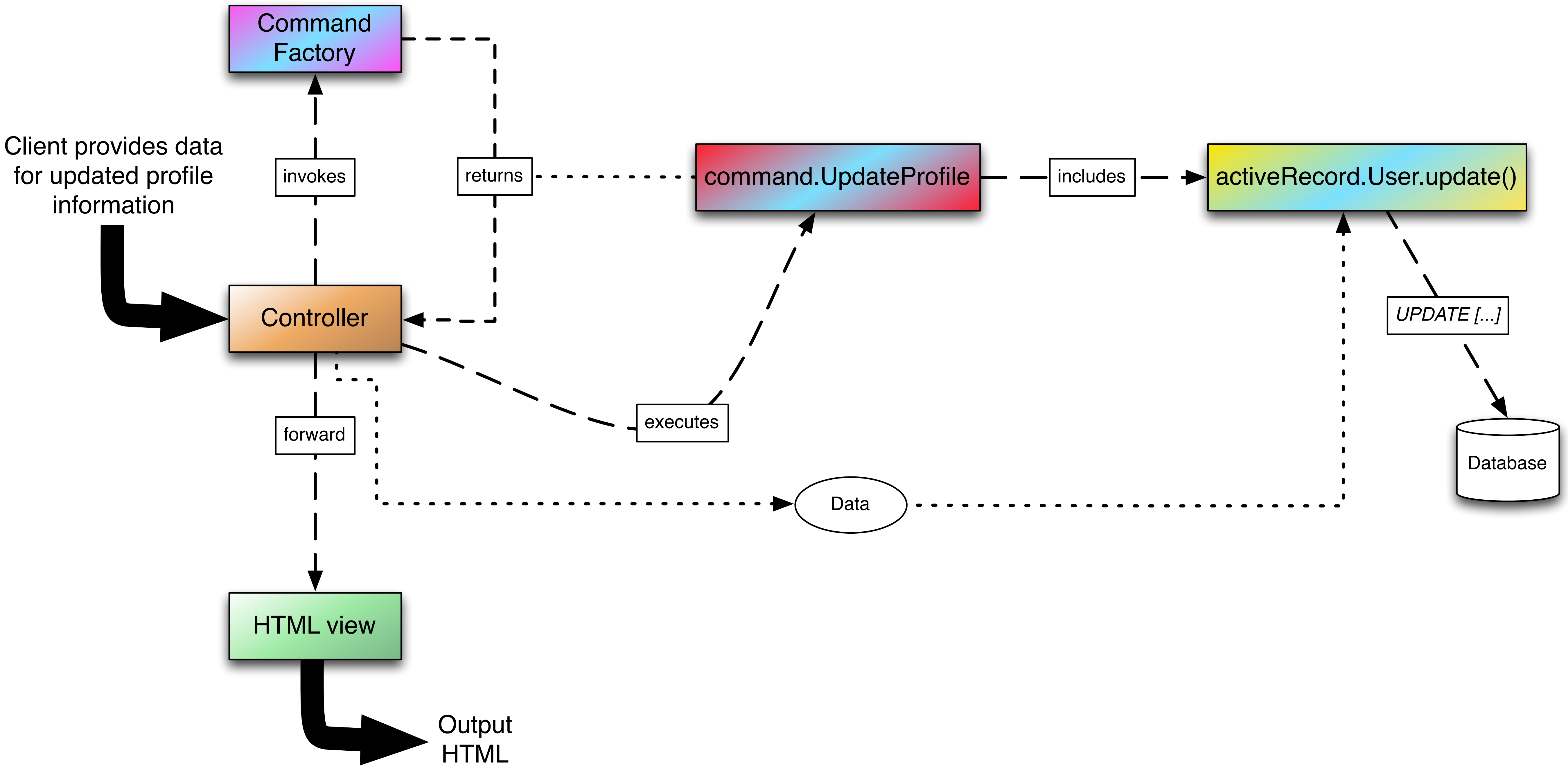
Key:

- Controller
- Model
- Factory Pattern
- Command Pattern
- Active Record Pattern
- View

Direct association:

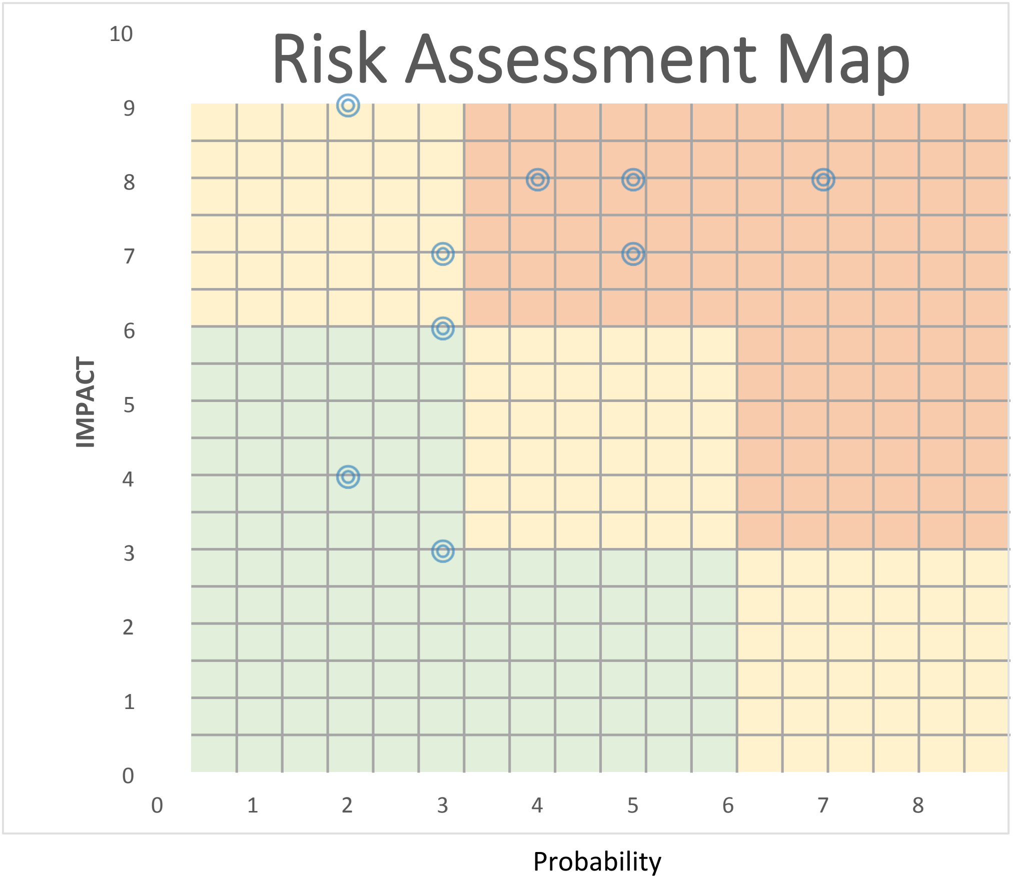
Indrect association:

SQL statement



Risk Assessment Table

Hazard	Avoidance actions	Reduction actions	Acceptance actions	Probability	Impact
Insufficient time	Stick to plan Prioritise important tasks.	None	Not acceptable; must be prevented and will be monitored.	7	8
Failure of the development hardware	Monitor health status of HW	None	Secondary equipment available.	5	8
Loss of data	Everyday backup.	Constant sync with cloud.	Not acceptable.	4	8
Physical destruction of all backups. (E.g. natural catastrophy)	Distribute backup to different locations.	None.	Not acceptable.	2	9
"Failure by design"	Get exptert opinion on planing.	Increase planning time.	Decrease functionality or non functional parts of project.	3	7
Problems while presentation.	Test presentation bevor actual presentation.	Good preperation for the presentation.	Not acceptable.	5	7
Additional functionality during programming phase.	None.	Build program modular.	No inclusion of the additional features.	2	4
Inhouse standards not met	Care for customer standards early	Verify whether adjustments possible	Low probability thus actions are taken at short notice	3	6
Corruption of a single file.	Backup of all files.	None.	Rebuild of the single file.	3	3


Risk Assessment Map
IMPACT
10
9
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
8
Probability